Problem 3-12: Polar Decomposition of Matrices

In class we have seen various product decompositions/factorizations of matrices like the LU-decomposition [Lecture → Section 2.3.2], the QR-decomposition [Lecture → Section 3.3.3], and the singular-value decomposition (SVD) [Lecture → Section 3.4]. In this problems we study another factorization, which is sometimes used in numerical methods, though it is not as important as the decompositions listed before.

This problem is related to [Lecture → Section 3.3.3.2] and [Lecture → Section 3.4.2] and requires familiarity with the EIGEN-based C++ implementation discussed in those sections.

Theorem 3.12.1. Polar decomposition

For every matrix $\mathbf{X} \in \mathbb{R}^{m,n}$, $m \geq n$, there is a matrix $\mathbf{Q} \in \mathbb{R}^{m,n}$ with orthonormal columns, $\mathbf{Q}^\top \mathbf{Q} = \mathbf{I}_n$, and a symmetric positive semi-definite [Lecture → Def. 1.1.2.6] matrix $\mathbf{M} \in \mathbb{R}^{n,n}$ such that $\mathbf{X} = \mathbf{QM}$.

The matrix factorization postulated in Thm. 3.12.1 is called **polar decomposition** of $\mathbf{X}$.

(3-12.a) ⚁ (30 min.) Give a proof of Thm. 3.12.1.

HIDDEN HINT 1 for (3-12.a) → `3-12-1-0:h1p.pdf`

SOLUTION for (3-12.a) → `3-12-1-1:s1.pdf` ▲

Based on the data types of EIGEN, the polar decomposition of a "tall/slim" real matrix is to be implemented as the following C++ class.

```
class PolarDecomposition {
 public:
  explicit PolarDecomposition(const Eigen::MatrixXd &X) { initialize(X); }
  PolarDecomposition(const Eigen::MatrixXd &A, const Eigen::MatrixXd &B);
  PolarDecomposition(const PolarDecomposition &) = default;
  ~PolarDecomposition() = default;

  // Left multiplication of M with the Q-factor of the polar decomposition
  void applyQ(Eigen::MatrixXd &Y) const { Y.applyOnTheLeft(Q_); }
  // Left multiplication of M with the M-factor of the polar decomposition
  void applyM(Eigen::MatrixXd &Y) const { Y.applyOnTheLeft(M_); }

 private:
  void initialize(const Eigen::MatrixXd &X);
  Eigen::MatrixXd Q_;  // factor Q
  Eigen::MatrixXd M_;  // factor M
};
```

The following specification of the class if given:

- The constructor

  ```
  PolarDecomposition(const Eigen::MatrixXd &X);
  ```

  should compute the polar decomposition factors $\mathbf{Q}$ and $\mathbf{M}$ according to Thm. 3.12.1 of the matrix passed in `X` and store them in the data members `Q_` and `M_`.

- The constructor

```
PolarDecomposition(const Eigen::MatrixXd &A, const
   Eigen::Matrix &B);
```

is supposed the initialize the data members `Q_` and `M_` with the polar decomposition factors $\mathbf{Q} \in \mathbb{R}^{m,n}$ and $\mathbf{M} \in \mathbb{R}^{n,n}$ of the matrix $\mathbf{A}\mathbf{B}^\top \in \mathbb{R}^{m,n}$, where the matrices $\mathbf{A} \in \mathbb{R}^{m,k}$ and $\mathbf{B} \in \mathbb{R}^{n,k}$ are passed through the arguments `A` and `B`.

- The methods `applyQ()` and `applyM()` realize the operations

$$\mathbf{Y} \leftarrow \mathbf{Q}\mathbf{Y} \quad , \quad \mathbf{Y} \leftarrow \mathbf{M}\mathbf{Y} \, ,$$

  where $\mathbf{Q}$ and $\mathbf{M}$ are the factors of the polar decomposition stored in the **PolarDecomposition** object.

(3-12.b) (15 min.) Regardless of the implementation, what is the *minimal* asymptotic computational cost, that is, a *sharp lower bound* of the asymptotic computational effort, for a call of the second constructor

```
PolarDecomposition(const Eigen::MatrixXd &A, const Eigen::Matrix
   &B);
```

of a **PolarDecomposition** object for a $m \times n$-matrix, $m \geq n$, for $m, n \to \infty$, and small fixed k?

$$\text{Minimal asymptotic cost} = O\left(\boxed{}\right) \quad \text{for} \quad m, n \to \infty \, .$$

SOLUTION for (3-12.b) → `3-12-2-0:.pdf` ▲

(3-12.c) (30 min.) [depends on Sub-problem (3-12.a)]

In the file `polardecomposition.hpp` implement the method

```
void PolarDecomposition::initialize(const Eigen::MatrixXd &X);
```

that sets the data members `_Q` and `_M` of the class **PolarDecomposition**. These data members store the factors $\mathbf{Q}$ and $\mathbf{M}$ of the polar decomposition of the argument matrix `X` according to Thm. 3.12.1.

HIDDEN HINT 1 for (3-12.c) → `3-12-3-0:pds2h.pdf`

SOLUTION for (3-12.c) → `3-12-3-1:s2.pdf` ▲

(3-12.d) (120 min.) [depends on Sub-problem (3-12.b)]

Write a code for the constructor

```
PolarDecomposition(const Eigen::MatrixXd &A,
                   const Eigen::Matrix &B);
```

which is supposed to initialize the data members `Q_` and `M_` with the polar decomposition factors $\mathbf{Q} \in \mathbb{R}^{m,n}$ and $\mathbf{M} \in \mathbb{R}^{n,n}$ of the matrix $\mathbf{X} := \mathbf{A}\mathbf{B}^\top \in \mathbb{R}^{m,n}$, where the matrices $\mathbf{A} \in \mathbb{R}^{m,k}$ and $\mathbf{B} \in \mathbb{R}^{n,k}$ are passed through the arguments `A` and `B`. We assume that k is small and fixed and $k \leq n < m$. Your code should have *optimal complexity* with respect to $m, n \to \infty$.

HIDDEN HINT 1 for (3-12.d) → `3-12-4-0:pcs3h1.pdf`

SOLUTION for (3-12.d) → `3-12-4-1:s2.pdf` ▲

End Problem 3-12 , 195 min.