PLUDecomposition

```
- _decomposition_is_done: bool
- _determinant: double
- _P: RowMatrix<int>
- _LU: RealMatrix
```

```
+ PLUDecomposition(M: const RealMatrix&, generate_exceptions: bool = false)
+ <<const>> isCorrect(): bool
+ <<const>> determinant(): double
+ template <typename T> solveLinearSystem(B: const Matrix<T>&, X: Matrix<T>&,
                                          represent_solutions_as_columns: bool = true): bool
+ <<const>> clone(): PLUDecomposition*
```

FactorizedUnpivotedLUDecomposition

```
- _decomposition_is_done: bool
- _determinant: double
- _L: RealMatrix
- _U: RealMatrix
```

```
+ FactorizedUnpivotedLUDecomposition(M: const RealMatrix&, generate_exceptions: bool = false)
+ <<const>> isCorrect(): bool
+ <<const>> determinant(): double
+ <<const>> L(): const RealMatrix&
+ <<const>> U(): const RealMatrix&
+ template <typename T> solveLLinearSystem(B: const Matrix<T>&, Y: Matrix<T>&,
                                           represent_solutions_as_columns: bool = true): bool
+ template <typename T> solveULinearSystem(Y: const Matrix<T>&, X: Matrix<T>&,
                                           represent_solutions_as_columns: bool = true): bool
+ <<const>> clone(): FactorizedUnpivotedLUDecomposition*
```

SVDecomposition

```
- _decomposition_is_done: bool
- _product_of_singular_values: double
- _U: RealMatrix
- _S: RowMatrix<double>
- _V: RealMatrix
```

```
+ SVDecomposition(M: const RealMatrix&, generate_exceptions: bool = false)
+ <<const>> isCorrect(): bool
+ <<const>> conditionNumber(): double
+ <<const>> reciprocalConditionNumber(): double
+ <<const>> productOfSingularValues(): double
+ template <typename T> solveLinearSystem(B: const Matrix<T>&, X: Matrix<T>&,
                                          represent_solutions_as_columns: bool = true): bool
+ <<const>> clone(): SVDecomposition*
```