

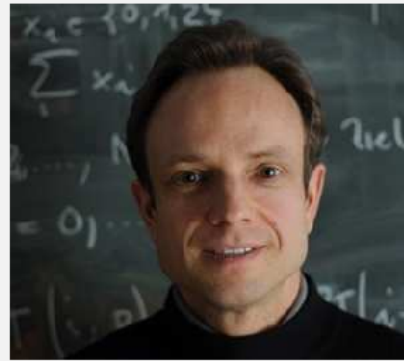
Course Video

Section 2.7.2: Sparse Matrices in EIGEN

Prof. R. Hiptmair, SAM, ETH Zurich

Date: September 18, 2020

(C) Seminar für Angewandte Mathematik, ETH Zürich



Dependencies. Section 1.2.1, Section 2.7.1

Duration: 12 minutes

II. Direct Methods for Square Linear Systems of Equations

2.7. Sparse Linear Systems

2.7.2. Sparse Matrices in Eigen

Standard: CRS/CCS format

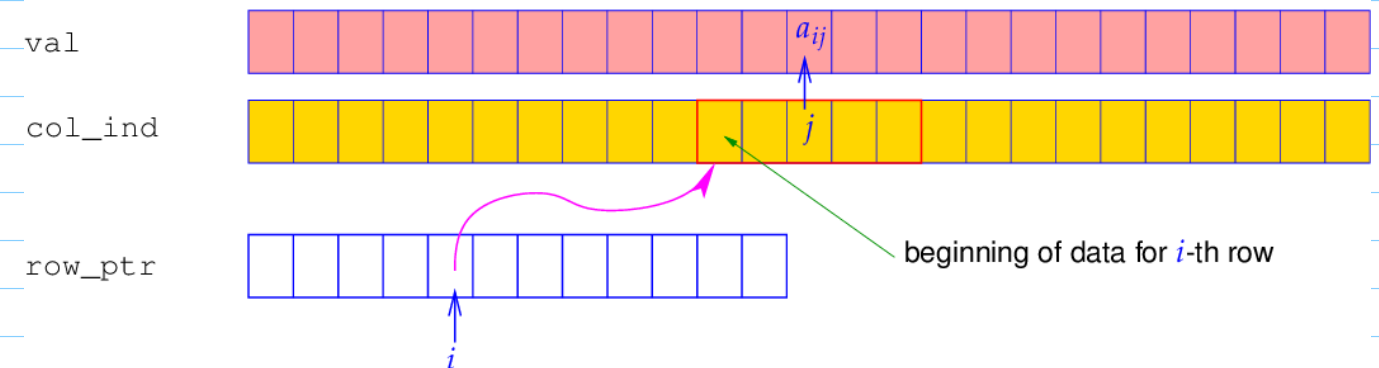
#include <Eigen/Sparse>

Eigen::SparseMatrix<int, Eigen::ColMajor> Asp(rows, cols); // CCS format

Eigen::SparseMatrix<double, Eigen::RowMajor> Bsp(rows, cols); // CRS format

Challenge: Efficient initialization

$$\text{val}[k] = a_{ij} \Leftrightarrow \begin{cases} \text{col_ind}[k] = j, \\ \text{row_ptr}[i] \leq k < \text{row_ptr}[i+1], \end{cases} \quad 1 \leq k \leq \text{nnz}(\mathbf{A}).$$



(2)

Set random entry $\triangleright$ massive data movement necessary

I. Idea: Use intermediate COO/triplet format

```
std::vector <Eigen::Triplet <double > > triplets;
// .. fill the std::vector triplets ..
Eigen::SparseMatrix<double, Eigen::RowMajor> spMat(rows, cols);
spMat.setFromTriplets(triplets.begin(), triplets.end());
```

$O(\# \text{ triplets})$ - complexity

II. reserve() & insert(), if nnz/per row,col. known

C++-code 2.7.2.1: Accessing entries of a sparse matrix: potentially inefficient!

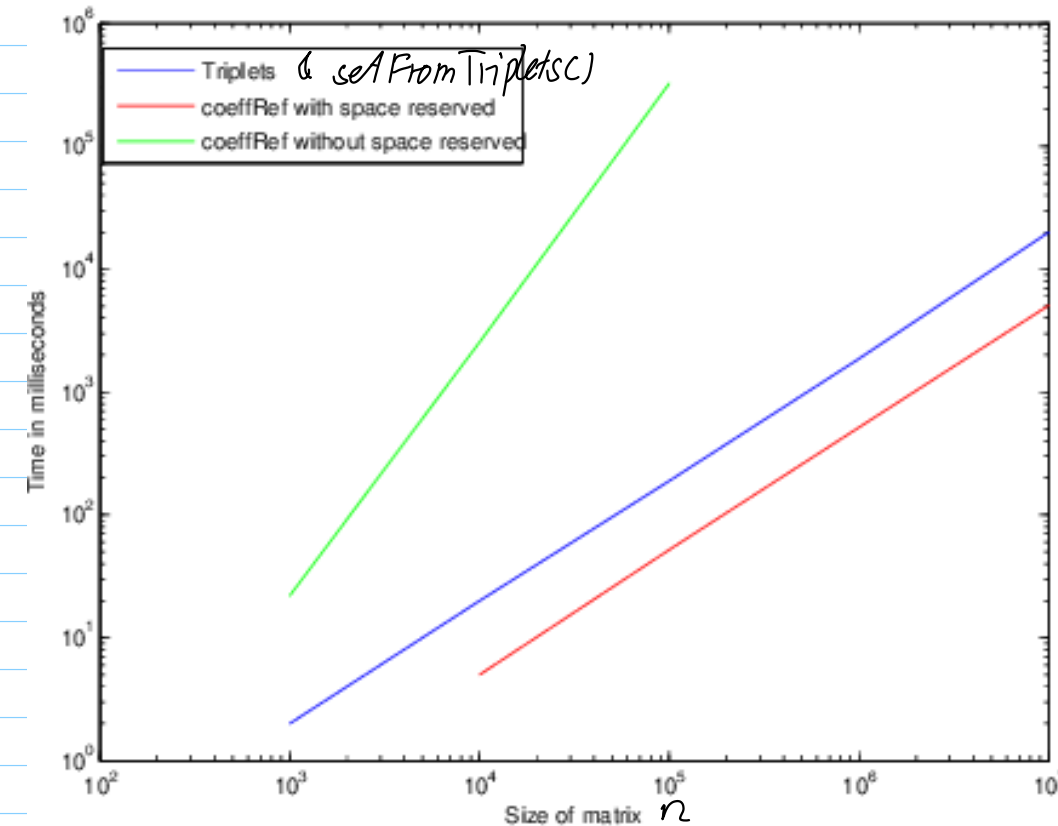
```
1 unsigned int rows, cols, max_no_nnz_per_row;
2 .....
3 SparseMatrix<double, RowMajor> mat(rows, cols);
4 mat.reserve(RowVectorXi::Constant(cols, max_no_nnz_per_row));  $\rightarrow$  Allocation of
5 // do many (incremental) initializations
6 for ( ) {
7   mat.insert(i, j) = value_ij;
8   mat.coeffRef(i, j) += increment_ij;
9 }
10 mat.makeCompressed();
```

$\} O(1)$, if enough space
reserve()'d

$\hookrightarrow$ squeeze out zeros

Exp 2.7.2.2: Initialization of SparseMatrix<>

$A \in \mathbb{R}^{n,n}$ banded, $(A)_{i,j} \neq 0 \Leftrightarrow |i-j| \leq 2$



$\} O(n)$

③

4

5

7