

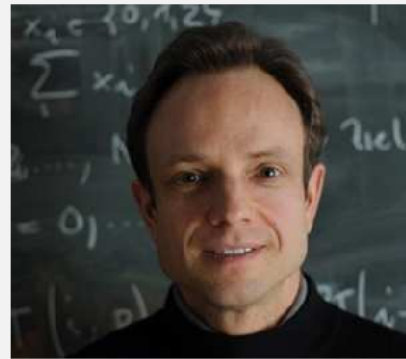
Course Video

Section 2.7.1: Sparse Matrix Storage Formats

Prof. R. Hiptmair, SAM, ETH Zurich

Date: September 18, 2020

(C) Seminar für Angewandte Mathematik, ETH Zürich



Dependencies. Section 2.1

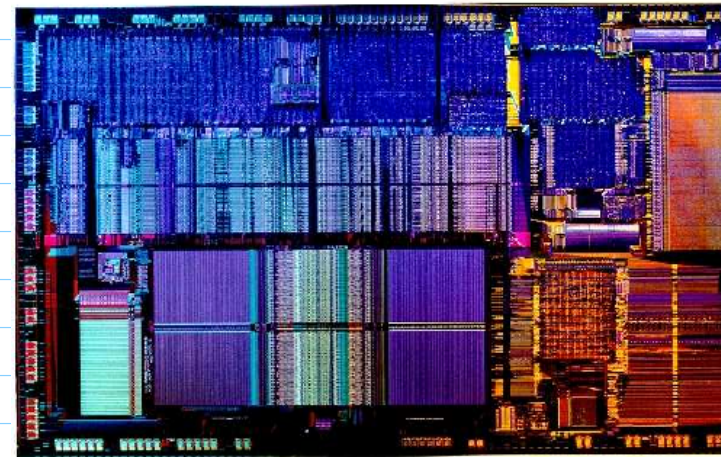
Duration: 20 minutes

II. Direct Methods for Square Linear Systems of Equations

2.7. Sparse Linear Systems

Sparse matrix : • "almost all" entries = 0
 • for which it is worth while exploiting this fact

Ex.: 2.7.0.4 : Sparse LSE in circuit simulation



• every is connected to only a few other nodes
 $\rightarrow$ LSE $Ax = b$
 $\text{nnz}(A) \sim \# \text{ nodes}$

$\triangleright$ Sparse LSE arise in models of large networks

② 2.7.1. Sparse Matrix Storage Formats

- Purposes :
- Memory $\sim \text{nnz}(A)$
 - $\text{cost}(A \times \text{vector}) \sim \text{nnz}(A)$
 - provide easily accessible information about location of non-zero entries

§ 2.7.1.1. COO/triplet format

→ Stores A as a list/sequence tuples (i, j, value)

```
struct Triplet {
    size_t i; // row index
    size_t j; // column index
    scalar_t a; // additive contribution to matrix entry
};
using TripletMatrix = std::vector<Triplet>;
```

C++-code 2.7.1.2: Matrix $\times$ vector product $y = Ax + y$ in triplet format

```
1 void multTripletMatvec(const TripletMatrix &A,
2                       const vector<scalar_t> &x,
3                       vector<scalar_t> &y)
4 for (size_t l=0; l<A.size(); l++) {
5     y[A[l].i] += A[l].a*x[A[l].j];
6 }
```

Allowed : repeated index pairs $\rightarrow$ values summed up

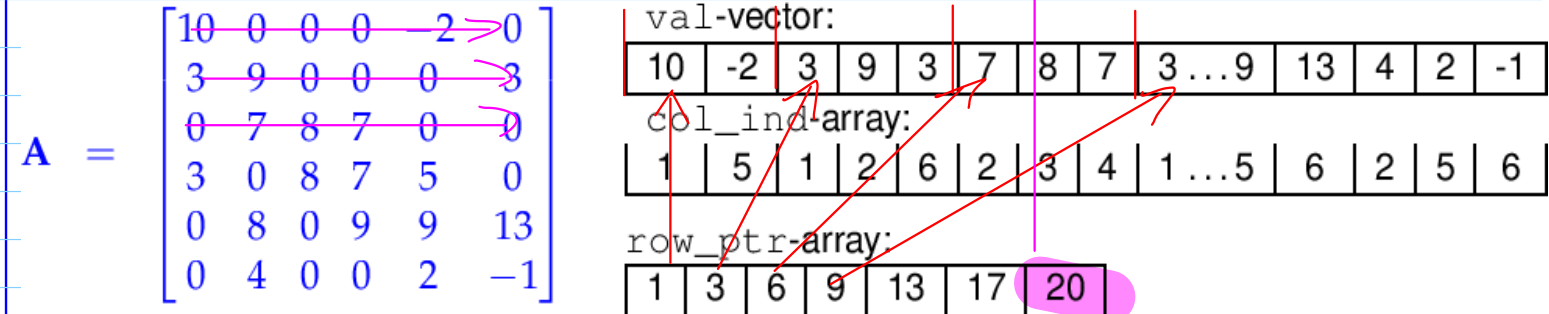
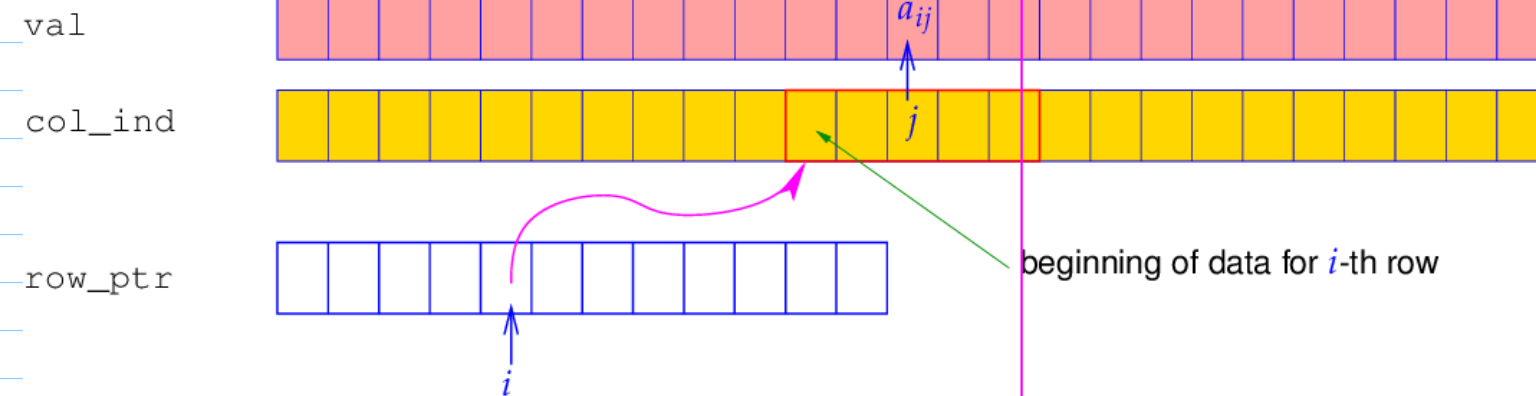
$$(A)_{ij} = \sum_{k: A[k].i=i, A[k].j=j} A[k].a$$

§ 2.7.1.4 : Compressed row storage (CRS)

$A \in \mathbb{K}^{n,n}$: stores non-zero entries, row-first

std::vector<scalar_t> val size $\text{nnz}(A) := \#\{(i,j) \in \{1,\dots,n\}^2, a_{ij} \neq 0\}$
 std::vector<size_t> col_ind size $\text{nnz}(A)$
 std::vector<size_t> row_ptr size $n+1$ & $\text{row_ptr}[n+1] = \text{nnz}(A) + 1$
 (sentinel value)

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



[LA indexing]

CRS: Non-zero entries reside in contiguous memory!

③

4

5

7