

# Rank Revealing QR Methods for Sparse Block Low Rank Solvers

---

**Esragul Korkmaz**, Mathieu Faverge, Grégoire Pichon, Pierre Ramet

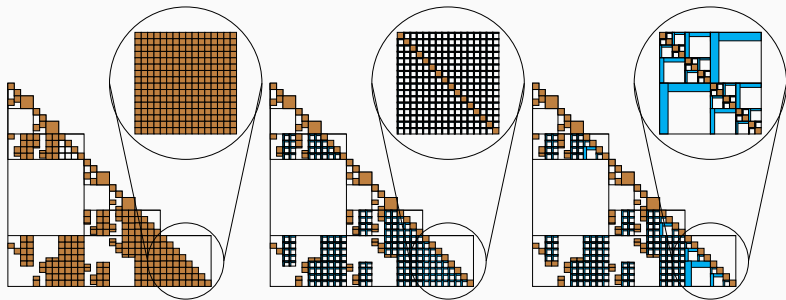
27 June 2019 – Compas

# Background

---

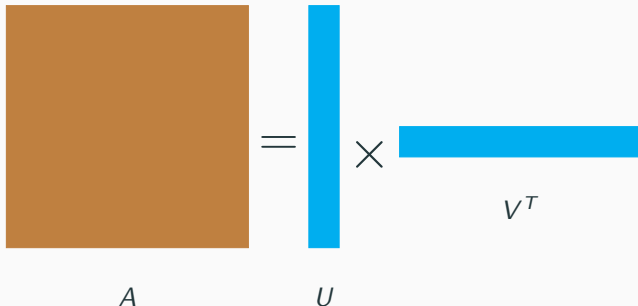
# Background - ANR SaSHiMi Project

## ANR SaSHiMi Project



- With: Mathieu Faverge, Pierre Ramet, Grégoire Pichon
- General Picture: Solve linear equations  $Ax=b$  for **large sparse systems**
- Full Rank Format: Too much memory usage
- Block Low Rank Format: Compression is possible, so less storage and faster
- Hierarchical Format: Even less computational complexity and memory consumption

# Background - Block Low Rank Structure



$$A \in \mathbb{R}^{m \times n}; U \in \mathbb{R}^{m \times r}; V \in \mathbb{R}^{n \times r}$$

- Compression reduces memory and cost of computations
- SVD is the most accurate with lowest rank:  $A_{m \times n} = \hat{U}_{m \times m} \Sigma_{m \times n} \hat{V}_{n \times n}$ 
  - $U = \hat{U}$  and  $V = \Sigma \hat{V}$
  - Too costly in practice
- In PASTIX, rank is decided through an automated way:  $\|A - UV^T\|_F \leq \epsilon \|A\|_F$

# Compression Methods

---

# Partial QR with Column Pivoting (PQRCP)

## Main Features

- Column pivoting: column with max 2-norm is the pivot
- $A = UV^T$  compression with column pivoting:
  - $AP = Q_{AP}R_{AP}$  is computed, where  $P$  is the permutation matrix
  - $U = Q_{AP}$  and  $V^T = R_{AP}P^T$
- All the trailing matrix is updated at each iteration

## Discussions

- 😐 Need larger rank than SVD for the same accuracy
- 😞 Not fast enough
- To reduce the cost of pivot selection
  - Randomized method with pivoting

# Randomized QR with Column Pivoting (RQRCP)

## Main Features

- Create independent and identically distributed Gaussian matrix  $\Omega$  of size  $b \times m$ , where  $b \ll m$
- Compute the sample matrix  $B = \Omega A$  of size  $b \times n$
- Find pivots on  $B$  where the row dimension is much smaller than  $A$ 
  - Less computations
  - Less communication
- Apply this pivoting to  $A$  like in PQRCP
- Update whole trailing matrix at each iteration like in PQRCP

## Discussions

- 😐 Similar accuracy to PQRCP
- 😞 Not fast enough
- To eliminate the cost of trailing matrix update:
  - Truncated randomized method with pivoting

# Truncated Randomized QR with Column Pivoting (TRQRCP)

## Main Features

- In RQRCP, only the current panel is updated at each iteration
  - Trailing matrix is not needed
- Extra storage: Reflector accumulations
- More efficient on large matrices with small ranks

## Discussions

- 😊 Fastest in sequential
- 😐 Similar accuracy to previous algorithms
- 😞 Can be improved to give closer ranks to SVD
- Instead of pivoting, apply a reasonable rotation to gather important information to the diagonal blocks
  - Randomized method with rotation



# Randomized QR with Rotation (RQRRT)

## Main Features

- Similar to RQRCP except:
  - Rotation applied to  $A$
  - Resampling
- In Randomized QR with Column Pivoting (RQRCP):
  - $BP_B = Q_B R_B$
  - $AP_B = Q_{AP} R_{AP}$
  - $U = Q_{AP}$  and  $V^T = R_{AP} P_B^T$
- In Randomized QR with Rotation (RQRRT):
  - $B^T = Q_B R_B$
  - $AQ_B = Q_{AQ} R_{AQ}$
  - $U = Q_{AQ}$  and  $V^T = R_{AQ} Q_B^T$

## Discussions

- 😊 Ranks closest to SVD
- 😞 Slower and updates whole trailing matrix each iteration

# Complexities

- **Blue:** No change, **Green:** Reduced cost, **Red:** More costly
- Matrix size  $n \times n$ , block size  $b$ , rank  $k$

Methods	Features
SVD: $\mathcal{O}(n^3)$	
PQRCP: $\mathcal{O}(n^2 k)$	pivot finding $\mathcal{O}(n^2)$ trailing matrix update $\mathcal{O}(n^2 k)$
PQRCP: $\mathcal{O}(n^2 k) \xrightarrow{\text{Randomization}} \text{RQRCP: } \mathcal{O}(n^2 k)$	sample matrix generation (beginning) $\mathcal{O}(n^2 b)$ pivot finding $\mathcal{O}(nb)$ update of sample matrix B $\mathcal{O}(nb^2)$ trailing matrix update $\mathcal{O}(n^2 k)$
RQRCP: $\mathcal{O}(n^2 k) \xrightarrow{\text{Truncation}} \text{TRQRCP: } \mathcal{O}(nk^2)$	sample matrix generation (beginning) $\mathcal{O}(n^2 b)$ pivot finding $\mathcal{O}(nb)$ update of current panel $\mathcal{O}(nk^2)$ update of sample matrix B $\mathcal{O}(nb^2)$
RQRCP: $\mathcal{O}(n^2 k) \xrightarrow{\text{Rotation}} \text{RQRRT: } \mathcal{O}(n^2 k)$	resampling (each iteration) $\mathcal{O}(n^2 b)$ rotation finding $\mathcal{O}(n^2 k)$ rotation of A $\mathcal{O}(n^2 k)$ trailing matrix update $\mathcal{O}(n^2 k)$

- Expected performances ( $>$  is better):  
 TQRCP  $>$  PQRCP  $>$  RQRCP  $>$  RQRRT  $>$  SVD

## Numerical Results

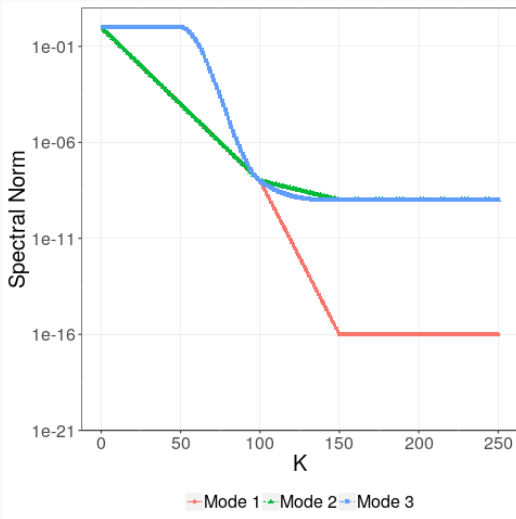
---

# Test Cases - Singular Values

For all Test Cases(Modes):

- Modes are different matrices
- Matrix Size 500
- Rank 100
- Generation Precision  $10^{-8}$
- $A = UDV^T$ 
  - D is diagonal matrix with singular values
  - U and V are orthonormal random matrices

Spectral Norms

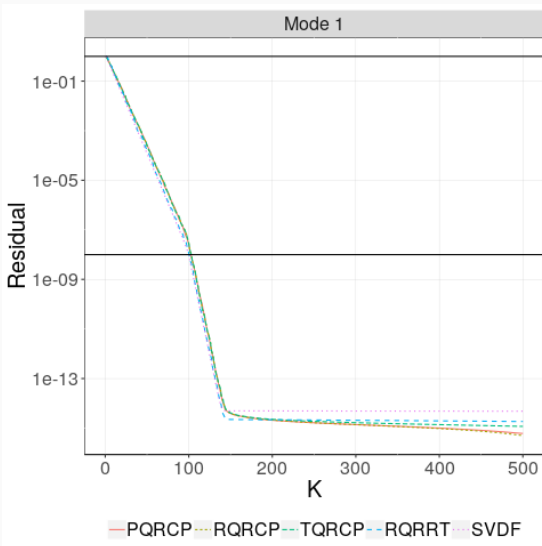


# Stability - First Test Case Residual Norms

For all Methods:

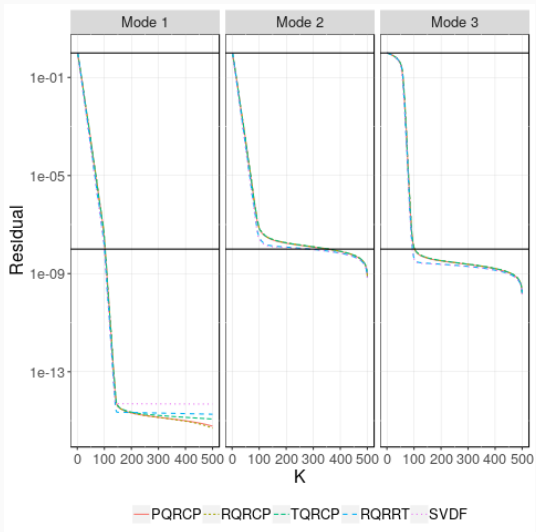
- Mode 1
- Matrix is fully factorized without any stopping criterion
- Residual:  $\frac{\|A - U_K V_K^T\|_F}{\|A\|_F}$
- K stands for index values of the matrix

Index vs Error



# Stability - All Test Cases Residual Norms

Index vs Error

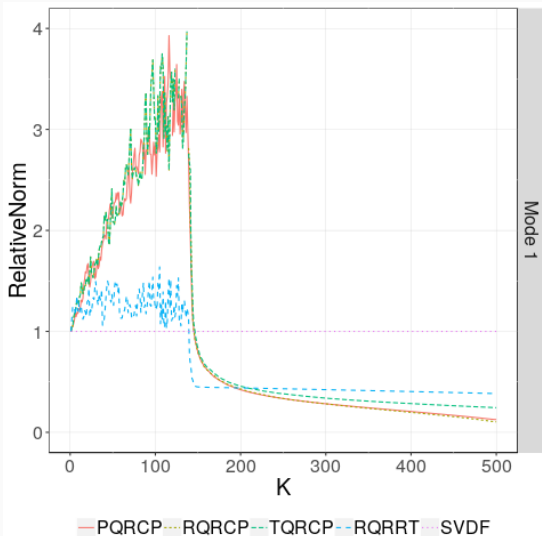


# Stability - First Test Case Relative Residual Norms

Index vs Relative Error

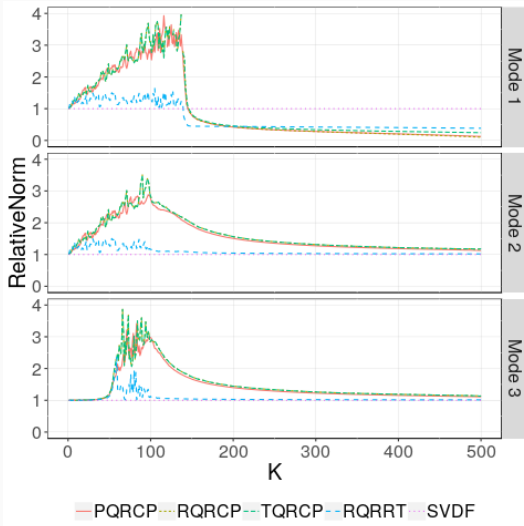
For all Methods:

- Matrix is fully factorized without any stopping criterion
- Relative Norm:  
$$\frac{\text{Residual}}{\text{Residual}^{\text{SVDf}}}$$
- K stands for index values of the matrix



# Stability - All Test Cases Relative Residual Norms

Index vs Relative Error



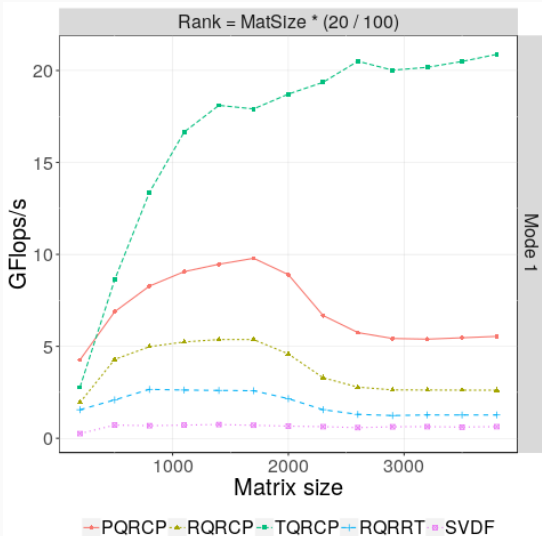


# Performance - First Test Case Gflops

## Matrix Size vs GFlops

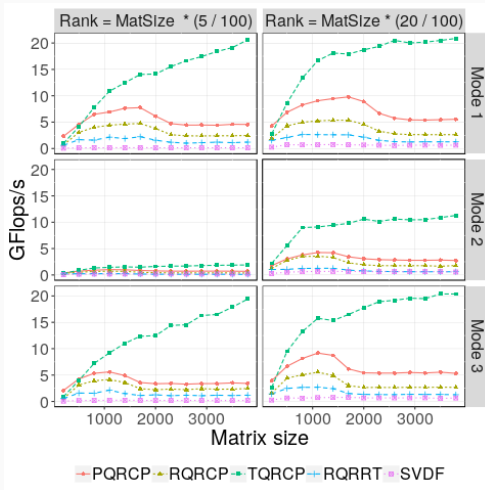
For all Methods:

- Mode 1
- $Rank = Matrix\_size * \frac{20}{100}$
- Different matrix sizes are checked
- Compression Precision  $10^{-8}$
- $GFlops = \frac{\min(GFlops)}{t}$
- Threshold is applied






# Performance - All Test Cases Gflops

## Matrix Size vs GFlops



- Application in the parallel framework of PASTIX with real, larger and tricky matrices
  - Why PQRCP has more performance than RQRCP?
  - TQRCP has the best performance for larger matrices with lower ranks but is not suitable for parallelism
  - RQRRT has the worst QR performance but is promising for parallel environment

-  Duersch, J. A.; Gu, M. (2017). "Randomized QR with Column Pivoting".
-  Xiao, J.; Gu, M.; Langou, J. (2017). "Fast Parallel Randomized QR with Column Pivoting Algorithms for Reliable Low-rank Matrix Approximations".
-  Martinsson, P. G. (2015). "Blocked Rank-revealing QR Factorizations: how randomized sampling can be used to avoid single-vector pivoting".

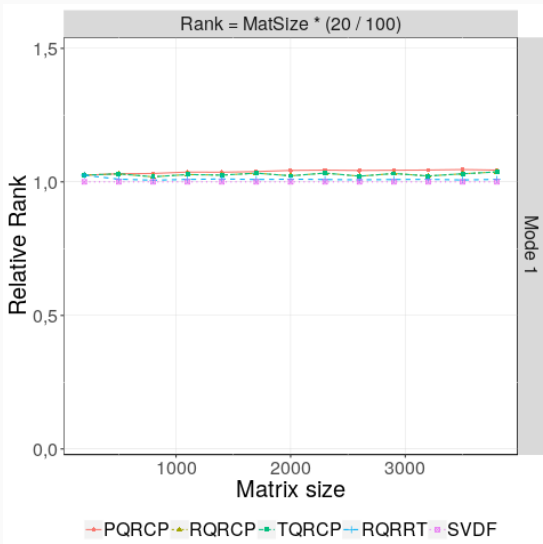
*THANK YOU!*

# Compression Ranks - First Test Case Relative Rank

Matrix Size vs Relative Rank

For all Methods:

- Mode 1
- $Rank = Matrix\_size * \frac{20}{100}$
- Different matrix sizes are checked
- Compression Precision  $10^{-8}$
- $RelativeRank = \frac{comp\_rank}{comp\_rank^{SVDF}}$
- Threshold is applied



# Compression Ranks - All Test Cases Relative Rank

Matrix Size vs Relative Rank

