A fast sparse QR factorization for solving linear least squares problems in graphics

Abeynaya Gnanasekaran
abeynaya@stanford.edu
Stanford University
USA

Eric Darve
darve@stanford.edu
Stanford University
USA

ABSTRACT

A wide range of problems in computer graphics and vision can be formulated as sparse least squares problems. For example, Laplacian mesh deformation, Least Squares Conformal Maps, Poisson image editing, and as-rigid-as-possible (ARAP) image warping involve solving a linear or non-linear sparse least squares problem. High performance is crucial in many of these applications for interactive user feedback. For these applications, we show that the matrices produced by factorization methods such as QR have a special structure: the off-diagonal blocks are low-rank. We leverage this property to produce a fast sparse approximate QR factorization, spaQR, for these matrices in near-linear time. In our benchmarks, spaQR shows up to 57% improvement over solving the normal equations using Cholesky and 63% improvement over a standard preconditioner with Conjugate Gradient Least Squares (CGLS).

ACM Reference Format:
Abeynaya Gnanasekaran and Eric Darve. 2021. A fast sparse QR factorization for solving linear least squares problems in graphics. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Talks (SIGGRAPH '21 Talks), August 09-13, 2021.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3450623.3464639

1 INTRODUCTION

The spaQR (sparsified QR) algorithm from [Gnanasekaran and Darve 2021] is built on the idea that certain off-diagonal blocks in A and A^TA of certain matrices are numerically low rank. By leveraging this property, spaQR produces a fast, sparse QR factorization of a full-rank, sparse matrix $A \in \mathbb{R}^{M\times N},\ M \geq N$ in $O(M \log N)$ time with $O(M)$ memory requirements. This is achieved at the expense of a small and controllable approximation error ϵ. The near-linear scaling with problem size and fast convergence when used as a preconditioner with CGLS make spaQR a high-performance solver for solving large, sparse least squares problems.

2 FAST QR FACTORIZATION FOR GRAPHICS

We demonstrate the superior performance of spaQR algorithm in solving least squares problems that come from three different applications in graphics. The test models were obtained from the collection of (textured) 3D models from the MIT CSAIL database and Stanford 3D Scanning repository. Convergence to a relative residual, $\|A^T(Ax-b)\|/\|A^Tb\| \leq 10^{-5}$ is noted for preconditioned CGLS with spaQR and a diagonal preconditioner.

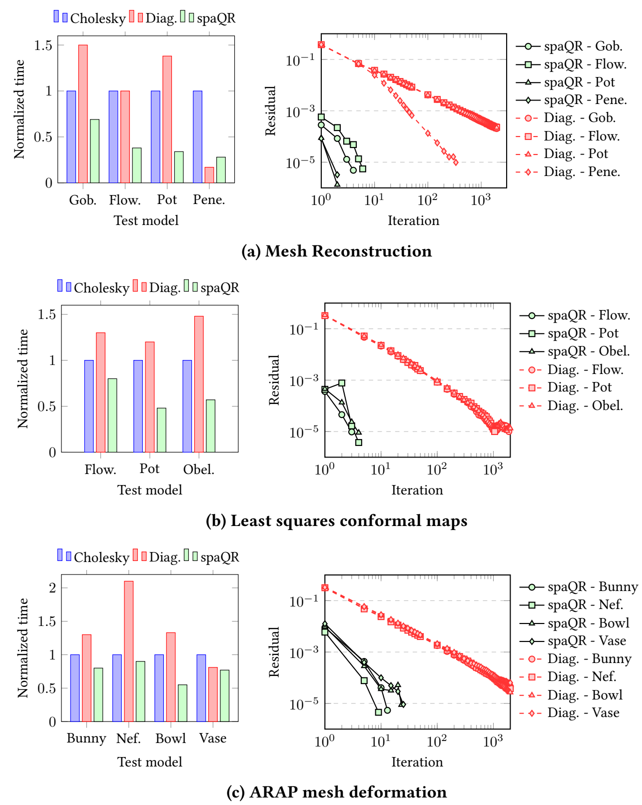

Figure 1: The plots on the left show the time-to-solution of spaQR and the diagonal preconditioner normalized with the time to solve the normal equations using Cholesky ('Normalized time'), while the ones on the right show the rate of convergence of preconditioned CGLS with spaQR and the standard diagonal preconditioner.

2.1 Mesh Reconstruction

Least-squares meshes [Sorkine and Cohen-Or 2004] is a technique that is used to reconstruct a mesh based on a sparse set of control points. It involves solving the least squares problem, $\min_x \|Lx\|^2 + \sum_{s\in C}(x_s - x_s')^2$, where L is the weighted graph Laplacian of the given mesh and x_s and x_s' are the positions of the control points C in the original mesh and reconstructed mesh respectively. This technique has numerous applications in graphics such as mesh

SIGGRAPH '21 Talks, August 09-13, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8373-8/21/08.
https://doi.org/10.1145/3450623.3464639

Table 1: Performance of spaQR for mesh reconstruction, LSCM, and ARAP mesh deformation in terms of the time to partition (t_P), factorize (t_F), and solve (t_S), number of iters. (#CGLS), and scaling of fact. time with problem size ($\frac{t_F}{M \log N}$).

Matrix	M (rows)	N (cols)	t_P (s)	t_F (s)	t_s (s)	#CGLS	$\frac{t_F}{M \log N}$ ($\times 10^{-6}$)
Goblet	339k	226k	3	7	1	4	3.8
Flowerpot	650k	500k	7	15	2	6	4.0
Pot	990k	762k	10	30	4	2	5.1
Penelope	2.6M	2M	47	59	14	2	3.6

(a) Mesh Reconstruction

Matrix	M (rows)	N (cols)	t_P (s)	t_F (s)	t_s (s)	#CGLS	$\frac{t_F}{M \log N}$ ($\times 10^{-6}$)
Flowerpot	2M	1M	12	44	4	3	3.6
Pot	3M	1.5M	27	65	9	4	3.6
Obelisk	3.2M	1.6M	18	71	10	4	3.6

(b) Least Squares Conformal Maps

Matrix	M (rows)	N (cols)	t_P (s)	t_F (s)	t_s (s)	#CGLS	$\frac{t_F}{M \log N}$ ($\times 10^{-5}$)
Bunny	1.8M	626k	11	83	15	13	2.1
Neferreti	2.7M	900k	14	123	19	11	2.1
Bowl	4.2M	1.4M	25	182	65	23	2.0
Vase	4.5M	1.5M	30	196	43	25	2.0

(c) ARAP Mesh Deformation

editing, shape modelling, smoothing, and sharpening. Figure 1a shows spaQR is on average 57% faster than Cholesky and 63% faster than the diagonal preconditioner (except on Penelope). As a preconditioner with CGLS, spaQR provides exponential convergence.

2.2 Least squares conformal maps (LSCM)

LSCM is a technique that is based on the least squares approximation of the Cauchy-Riemann equations and is used to map a 3D surface onto a 2D surface such that angles are preserved locally [Levy et al. 2002]. Conformal maps are invaluable in various applications like remeshing, texture mapping, 3D fabrication, and shape analysis. Figure 1b shows that spaQR outperforms the standard solvers with an improvement of 38% over Cholesky and 53% over diagonal preconditioner.

2.3 ARAP mesh deformation

ARAP mesh deformation [Sorkine and Alexa 2007] is used to interactively edit 3D meshes by minimizing a warping energy. The energy formulation is non-linear and penalizes deviations from local rigidity while satisfying a set of user-defined constraints. The non-linear energy is minimized by iteratively solving a linear least squares problem. We test our algorithm on one such linear least squares problem. Figure 1c shows that the spaQR is 25% faster than Cholesky and 51% faster than the diagonal preconditioner.

Overall, spaQR performs much better than the standard techniques in the three applications. The time of factorization is a fraction of the time for Cholesky factorization. The spaQR preconditioned CGLS shows exponential convergence in all of the tests. Results in Table 1 show that the time to factorize the matrix scales as $O(M \log N)$ (see last column).

Figure 2: The matrix block A_p (left), low-rank approximation of the off-diagonal blocks (middle), sparsified matrix (right). The red blocks are 'small' ($O(\epsilon)$) and are ignored leading to a smaller and sparser matrix.

3 METHOD

The spaQR algorithm, is built on top of a Nested Dissection based sparse multifrontal QR factorization. The key idea in the algorithm is to continually decrease the size of the nested dissection separators by using a low-rank approximation of its neighbors in the graph of $A^T A$. Let p be a connected subset of a separator and let n be the set of all rows / columns such that A_{np} and / or A_{pn} is non-zero. Considering only a small piece of the matrix and focusing on the non-zero blocks for p and n we have:

$$A_p = \begin{bmatrix} A_{pp} & A_{pn} \\ A_{np} & A_{nn} \end{bmatrix} = \begin{bmatrix} I & A_{p_1 n} \\ 0 & A_{p_2 n} \\ A_{np} & A_{nn} \end{bmatrix} \qquad A_{pp} = \begin{bmatrix} I \\ 0 \end{bmatrix}$$

where $A_{pp} \in \mathbb{R}^{r_p \times c_p}$ with $r_p \geq c_p$; we perform a block diagonal scaling to get every A_{pp} diagonal block in this form.

The main assumption in the spaQR algorithm is that the off-diagonal blocks ($A_{p_1 n}$, $A_{p_2 n}$, A_{np}) are low-rank. We compute a low-rank approximation of the off-diagonal blocks by using rank revealing QR or singular value decomposition. Then, by dropping the singular values below a threshold, ϵ, we can reduce the size of the blocks. Figure 2 shows this sparsification process on the block matrix A_p. By ignoring the $O(\epsilon)$ blocks, we can reduce the cost of the QR factorization and achieve the $O(M \log N)$ complexity.

4 CONCLUSION

We showed that spaQR is an efficient solver for sparse least squares problems in graphics by focusing on three applications. We also found that spaQR improves the performance in 2D image warping and Poisson image editing applications. The factorization time scales linearly with the problem size and has exponential convergence when used as a preconditioner with CGLS. Furthermore, the structure of the spaQR algorithm exhibits more parallelism than direct methods which is immensely useful for real-time performance on GPUs in interactive applications. Finally, the code is freely available for use at `https://github.com/Abeynaya/spaQR_public`.

REFERENCES

Abeynaya Gnanasekaran and Eric Darve. 2021. Hierarchical Orthogonal Factorization: Sparse Least Squares Problems. arXiv:2102.09878 [math.NA]

Bruno Levy, Sylvain Petitjean, Nicolas Ray, and Jérôme Maillot. 2002. Least Squares Conformal Maps for Automatic Texture Atlas Generation. *ACM Trans. Graph.* 21 (07 2002), 362–371. https://doi.org/10.1145/566654.566590

Olga Sorkine and Marc Alexa. 2007. As-Rigid-as-Possible Surface Modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing* (Barcelona, Spain) *(SGP '07)*. Eurographics Association, Goslar, DEU, 109–116.

O. Sorkine and D. Cohen-Or. 2004. Least-squares meshes. In *Proceedings Shape Modeling Applications, 2004.* 191–199. https://doi.org/10.1109/SMI.2004.1314506