Warm start in Quadratic Programming solver

When solving practical problems, there's often a need to address not only individual problem but sequence where each subsequent task differs only slightly from the previous one. This underscores the need for efficient methods to expedite the solution process, such as "warm starting" — leveraging results or data from previous tasks to facilitate quicker starts on subsequent ones.

In the context of quadratic programming methods, particular attention is given to stages related to Cholesky factorization and LU decomposition, which are often the most time-consuming. When minor changes occur in the matrix of the next task, the challenge arises of efficiently updating the Cholesky factorization for a Symmetric Positive Definite (SPD) matrix. This becomes especially relevant when k rows and their corresponding k columns are removed from the matrix.

Therefore, to optimize and accelerate the solution process for series of tasks, it's crucial to develop and employ methods that not only allow for the effective utilization of results from previous solutions but also ensure swift updates and recalculations of the necessary matrices and factorizations.

Cholesky and LU factorizations are fundamental techniques in numerical linear algebra, crucial for solving linear systems. Traditionally, the removal of rows and columns from a matrix necessitates a complete refactorization, which can be computationally expensive, especially for large matrices.

The works by Gondzio, "Stable Algorithm for Updating Dense LU Factorization after Row or Column Exchange and Row and Column Addition or Deletion," and Devis, "Row Modifications of a Sparse Cholesky Factorization," offer valuable insights into dynamically updating these factorizations, significantly reducing the need for a complete recomputation. However, these papers primarily focus on the scenario of rank-1 updates.

The challenge presented here seeks to build upon these fundamental insights and address more complex situations where k rows and the corresponding k columns are removed. This exploration into the domain of higher-dimensional updates aims not only to boost computational efficiency and diminish the overhead linked to such matrix modifications but also to expand the utility of these essential factorization techniques in accommodating substantial alterations in the matrix structure.

Within the scope of this challenge, we are keen to gain comprehensive insights into various facets of the topic. These can encompass, but are not limited to:

1. Literature Review: This component entails a comprehensive review of existing literature, emphasizing mathematical and algorithmic strategies for updating Cholesky and LU factorizations. The review will identify key methodologies, their strengths, weaknesses, and applications in the context of matrix modifications and updates.
2. Algorithm Development: In this phase, the objective is to devise and implement an innovative algorithm that incorporates the identified techniques for dynamically updating factorizations when k rows and corresponding columns are removed. The algorithm's versatility is paramount; it should be adaptable to both dense and sparse matrices, ensuring efficient and accurate factorization updates.
3. Performance Evaluation: This segment necessitates a rigorous evaluation of the developed algorithm's performance. It involves benchmarking the computational efficiency, expressed

through time complexity metrics, and assessing its numerical stability. Comparative analyses against conventional refactorization methods will be conducted across various matrix dimensions, different values of k, and varying degrees of matrix sparsity to ascertain the algorithm's robustness and effectiveness.

4. Optimality Conditions: The final task centers on establishing the optimality conditions for employing the update method over complete refactorization. This includes investigating the influence of parameters like the magnitude of k, matrix properties, and others on the efficiency and effectiveness of the update approach. A comprehensive discussion on the potential trade-offs between the update method and traditional refactorization techniques will be pivotal in understanding the scenarios where the update method offers a distinct advantage.

We would greatly appreciate any insights, recommendations, or innovative approaches that could assist in addressing this challenge. Your expertise and contributions could significantly enhance our collaborative efforts, paving the way for deeper exploration and fruitful future partnerships.

References

1) Jacek Gondzio, "Stable Algorithm for Updating Dense LU Factorization after Row or Column Exchange and Row and Column Addition or Deletion," Optimization, vol. 23, no. 1, pp. 1-18, 1992.
2) Timothy A. Davis and William W. Hager, "Row Modifications of a Sparse Cholesky Factorization," SIAM Journal on Matrix Analysis and Applications, vol. 26, no. 3, pp. 621-637, 2005.
3) Timothy A. Davis and William W. Hager, "Modifying a Sparse Cholesky Factorization," SIAM Journal on Matrix Analysis and Applications, vol. 20, no. 3, pp. 606-627, 1999.
4) Timothy A. Davis and William W. Hager, "Multiple-Rank Modifications of a Sparse Cholesky Factorization," SIAM Journal on Matrix Analysis and Applications, vol. 22, no. 4, pp. 997-1013, 2000.
5) Robert B. Schnabel and Elizabeth Eskow, "A Revised Modified Cholesky Factorization Algorithm," SIAM Journal on Optimization, vol. 9, no. 4, pp. 1135-1148, 1999.