A SPARSE IMPLEMENTATION OF THE FRISCH-NEWTON ALGORITHM FOR QUANTILE REGRESSION

ROGER KOENKER AND PIN NG

Abstract. Recent experience has shown that interior-point methods using a log barrier approach are far superior to classical simplex methods for computing solutions to large parametric quantile regression problems. In many large empirical applications, the design matrix has a very sparse structure. A typical example is the classical fixed-effect model for panel data where the parametric dimension of the model can be quite large, but the number of non-zero elements is quite small. Adopting recent developments in sparse linear algebra we introduce a modified version of the Frisch-Newton algorithm for quantile regression described in Koenker and Portnoy (1997). The new algorithm substantially reduces the storage (memory) requirements and increases computational speed. The modified algorithm also facilitates the development of nonparametric quantile regression methods. The pseudo design matrices employed in nonparametric quantile regression smoothing are inherently sparse in both the fidelity and roughness penalty components. Exploiting the sparse structure of these problems opens up a whole range of new possibilities for multivariate smoothing on large data sets via ANOVA-type decomposition and partial linear models.

1. Introduction

Significant progress has been made over the last decade to improve the computational efficiency of interior point methods for linear programming. Primal-dual interior-point algorithms are competitive with classical simplex methods for small to moderate size problems and far superior for large problems. See Gonzaga (1992), Lustig, Marsten and Shanno (1994), and Wright (1997) for surveys of the development of interior point methods. Recent experience with quantile regression, described in Koenker and Portnoy (1997), has shown that primal-dual interior-point algorithms are competitive with least squares methods for large parametric quantile regression problems when the parametric dimension is moderate.

In many large empirical applications of quantile regressions, however, the parametric dimension of the model can be quite large. This can adversely affect the performance of some interior point implementations. Often though, such problems have a design matrix that has a very sparse structure. A typical example is the classical fixed-effect model for panel data where the parametric dimension of the model can be quite large, due to a large number of indicator variables and their interactions, but the number of non-zero elements in the design matrix can be quite small. The design matrices arising in nonparametric quantile regression smoothing problems are also inherently sparse in both their fidelity and roughness penalty components, see e.g., Koenker, Ng and Portnoy (1994), He, Ng and Portnoy (1998), He and Ng (1999) and Koenker and Mizera (2001).

In this paper we adapt recent developments in sparse linear algebra to introduce a modified version of the Frisch-Newton algorithm for quantile regression described in Koenker and Portnoy (1997). The new algorithm utilizes the BLAS-like routines for sparse matrices available from Saad (1994) and the block sparse Cholesky algorithm of Ng and Peyton (1993). Our algorithm substantially reduces storage (memory) requirements and increases computational speed. Exploiting the sparse structure opens up a whole range of new applications for multivariate smoothing on large data sets via ANOVA-type decomposition and partial linear models.

2. Quantiles Regression as a Linear Program

Given $\tau \in [0,1]$ and n observations on the dependent variable y_i and the p-variate independent variable x_i, the τ-th parametric linear regression quantile b is obtained by solving

$$\min_{b\in\mathbb{R}^p} \sum_{i=1}^{n} \rho_\tau(y_i - x_i'b) \tag{1}$$

where $\rho_\tau(u) = u(\tau - I(u<0))$. Letting e denote an n-vector of ones, we can rewrite (1) as the linear program:

$$\min_{(u',v',b')} \{\tau e'u + (1-\tau)e'v | Xb + u - v = y, \quad (u', v', b') \in \mathbb{R}^{2n}_{+} \times \mathbb{R}^p\} \tag{2}$$

Here, u and v denote the positive and negative parts of the regression residual. The dual of (2) is given by

$$\max_d \{y'd | X'd = (1-\tau)X'e, \quad d \in [0,1]^n\} \tag{3}$$

where $[0,1]^n$ denotes the n-fold Cartesian product of the unit interval, and d may be interpreted as a vector of Lagrange multipliers associated with the linear equality constraints of the primal problem. It is the dual linear program in (3) that we solve using the interior point methods.

2.1. **Notation.** In the remainder of the paper we will adopt the notational conventions of the numerical analysis literature as e.g., Lustig, Marsten and Shanno (1992). the primal-dual pair is described as

$$\begin{aligned} &\min_x \{c'x | Ax = b,\ 0 \le x \le u, \quad x \in \mathbb{R}^n\} \\ &\max_{(z,w,y)} \{b'y | A'y + z - w = c, \quad (z,w) \ge 0, \quad (y,z,w) \in \mathbb{R}^{2n}_{+} \times \mathbb{R}^p\}. \end{aligned} \tag{4}$$

Note that the dual linear program in (3) that we want to solve corresponds to the primal problem above in (4) so that $-y$, d, $(1-\tau)\,X'e$, and X' in (3) correspond, respectively, to c,x,b, and A in (4). Note also that we have generalized the problem slightly to allow u to be an arbitrary vector of upper bounds. This is convenient, for example, when we eventually deal with penalty functions in nonparametric regression problems.

3. The Frisch-Newton Algorithm

The log-barrier form of the Lagrangian for the primal problem (4) is,

$$L = c'x - y'(Ax-b) - w'(u-x-s) - \mu(\sum \log x_i + \sum \log s_i),$$

for $\mu > 0$. The final log-barrier term in the expression guarantees that (x,s) stays away from the boundary of the positive orthant. The strategy is to gradually relax μ, letting it tend toward zero as the duality gap $c'x - b'y > 0$ closes and we approach the optimal solution. Differentiating the Lagrangian with respect to x, y, w, s, equating to zero, and defining $z = \mu X^{-1}e$, the classical Karush-Kuhn-Tucker (KKT) conditions for optimality are,

$$g(\xi) = \begin{pmatrix} A'y + z - w - c \\ Ax - b \\ x + s - u \\ XZe - \mu e \\ SWe - \mu e \end{pmatrix} = 0 \tag{5}$$

where the upper case letters are diagonal matrices with the corresponding lower case vectors as their diagonal elements, so for example $X = \mathrm{diag}(x)$. Given an initial point $\xi_0 = (y_0, z_0, x_0, , s_0, w_0)$ the Newton approximation to (5) is

$$g(\xi) \approx \nabla_\xi g(\xi_0) d\xi + g(\xi_0). \tag{6}$$

We choose a descent direction $d\xi$ by setting this approximation equal zero, i.e.,

$$d\xi = -[\nabla_\xi g(\xi_k)]^{-1} g_k.$$

Frisch (1955) was a pioneering advocate of the log-barrier method for solving linear programming problems, so Portnoy and Koenker (1997) call their implementation of the log-barrier method for quantile regression a Frisch-Newton algorithm. The linear system we obtain from setting (6) equal zero looks like

$$\begin{bmatrix} A' & I & 0 & 0 & -I \\ 0 & 0 & A & 0 & 0 \\ 0 & 0 & I & I & 0 \\ 0 & X & Z & 0 & 0 \\ 0 & 0 & 0 & W & S \end{bmatrix} \begin{bmatrix} dy \\ dz \\ dx \\ ds \\ dw \end{bmatrix} = - \begin{bmatrix} A'y + z - w - c \\ Ax - b \\ x + s - u \\ XZe - \mu e \\ SWe - \mu e \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{bmatrix} \tag{7}$$

All interior-point algorithms involve generating sequences of the primal-dual variables and iterating until the duality gap becomes smaller than a specified tolerance. A critical aspect of this approach is the choice of an updating strategy for the barrier parameter μ.

The most successful updating strategy to date, is the predictor-corrector method of Mehrotra (1992). This is the approach adopted by Lustig, Marsten and Shanno (1994) and by Portnoy and Koenker (1997). At each iteration the algorithm computes an affine-scaling "predictor" direction, step lengths are then computed and μ is updated, and finally a "corrected" direction is computed and a step is taken. The affine-scaling predictor steps are given by solving 7 with $\mu = 0$,

$$\begin{aligned} dy &= (AQ^{-1}A')^{-1}[\tilde{r}_2 + AQ^{-1}\tilde{r}_1] \\ dx &= Q^{-1}(A'dy - \tilde{r}_1) \\ ds &= -dx \\ dz &= -z - X^{-1}Zdx = -Z\left(e + X^{-1}dx\right) \\ dw &= -w - S^{-1}Wds = -W\left(e + S^{-1}ds\right) \end{aligned} \tag{8}$$

where $Q = X^{-1}Z + S^{-1}W$, $\tilde{r}_1 = c - A'y$ and $\tilde{r}_2 = b - Ax$.

The maximum feasible affine-scaling primal and dual step lengths that ensure the primal and dual variables x, s, z and w stay feasible are then

$$\begin{aligned} \alpha_P &= \arg\max\{\alpha \in [0,1]\,|x + \alpha dx \geq 0, s + \alpha ds \geq 0\} \\ \alpha_D &= \arg\max\{\alpha \in [0,1]\,|z + \alpha dz \geq 0, w + \alpha dw \geq 0\} \end{aligned} \tag{9}$$

The duality gap from taking this affine-scaling step is

$$\mu_{as} = (x + \alpha_P dx)'(z + \alpha_D dz) + (s + \alpha_P ds)'(w + \alpha_D dw)$$

This new duality gap is then compared to the existing duality gap $\mu_0 = x'z + s'w$ to evaluate the amount of centering needed. If μ_{as} is much smaller than the current μ_0, it suggests that the affine-scaling step manages to reduce the duality gap significantly and, hence, not much centering is needed. Therefore, μ can be reduced substantially. On the other hand, if the difference between μ_{as} and μ_0 is small, the affine-scaling step will not lead to large improvement and, hence, substantial centering is needed. Mehrotra's centering proposal involves replacing μ in (7) by

$$\mu = \left(\frac{\mu_{as}}{\mu_k}\right)^2 \left(\frac{\mu_{as}}{n}\right) \tag{10}$$

Following Lustig, Marsten and Shanno (1992), the third component, corrector direction, $\delta\xi = (\delta y', \delta z', \delta x', \delta s', \delta w')$ are obtained by substituting $\xi = \xi + \delta\xi$ into (5), setting $r_1 = r_2 = r_3 = r_4 = r_5 = 0$ and solving for $\delta\xi$. The linear system obtained is similar to (7):

$$\begin{bmatrix} A' & I & 0 & 0 & -I \\ 0 & 0 & A & 0 & 0 \\ 0 & 0 & I & I & 0 \\ 0 & X & Z & 0 & 0 \\ 0 & 0 & 0 & W & S \end{bmatrix} \begin{bmatrix} \delta y \\ \delta z \\ \delta x \\ \delta s \\ \delta w \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ 0 \\ dXdZe - \mu e \\ dSdWe - \mu e \end{bmatrix} \tag{11}$$

Notice the main difference between the systems in (7) and (11) is the nonlinear terms $dXdZe$ and $dSdWe$ on the right-hand-side. The computational implication from this is critical because the major amount of computational resource is expended in the Cholesky factorization of the coefficient matrix in the left-hand-side, which only needs to be performed once for each iteration. Applying similar algebra for the affine-scaling predictor direction given in (8) yields the following expressions for the corrector steps:

$$\begin{aligned} \delta y &= (AQ^{-1}A')^{-1}[AQ^{-1}\hat{r}_1] \\ \delta x &= Q^{-1}(A'\delta y - \hat{r}_1) \\ \delta s &= -\delta x \\ \delta z &= -X^{-1}Z\delta x + X^{-1}(\mu e - dXdZe) \\ \delta w &= -S^{-1}W\delta s + S^{-1}(\mu e - dSdWe) \end{aligned} \tag{12}$$

where $\hat{r}_1 = \mu(S^{-1} - X^{-1})e + X^{-1}dXdZe - S^{-1}dSdWe$. The motivation for the corrector steps is given in Lustig, Marsten and Shanno (1992, pp. 441-2) and Wright (1997, pp.196-7). Basically, the correction is an attempt to correct for the amount of deviation, $dXdZ$, of XZ from its targeted zero values in (5) when taking the affine-scaling step.

4. Sparse Linear Algebra

In large problems almost all of the computational effort of the Frisch-Newton algorithm occurs in the solution of the linear system involving the matrix $AQ^{-1}A'$ matrix in (8) and (12). Fortuitously, both solutions can use the same Cholesky factorization so this operation only needs to be performed once for the affine-scaling step $d\xi$ and can be reused for the corrector steps $\delta\xi$ in each iteration. Recall that given the Cholesky factorization $LL' = A$, the symmetric, positive definite linear system $Ax = b$ can be solved in two steps by backsolving the triangular systems:

$$\begin{aligned} Ly &= b \\ L'x &= y \end{aligned}$$

The block sparse algorithm of Ng and Peyton (1993) provides an extremely efficient Cholesky factorization for solving a linear systems of equations with symmetric positive definite coefficient matrix. It consists of four distinct steps: (1) ordering, (2) symbolic factorization, (3) numerical factorization and (4) numerical solution. The ordering step attempts to reorder the $AQ^{-1}A'$ matrix using the multiple minimum degree routines from Liu (1985) to reduce the fill-in and the amount of work required by the factorization steps. Symbolic factorization generates the compact data structure in which the Cholesky factor L will be computed and stored. Numerical factorization computes the sparse Cholesky factor using the efficient data structures obtained from symbolic factorization. The numerical solution step merely performs the triangular eliminations needed to solve the linear system.

5. Application to Bivariate Smoothing

To obtain a better sense of the computational cost of Ng and Peyton's algorithm, we use the bivariate test function

$$f_0(x,y) = \frac{40\exp\left(8(x-.5)^2 + (y-.5)^2\right)}{\exp\left(8\left((x-.2)^2 + (y-.7)^2\right)\right) + \exp\left(8\left((x-.7)^2 + (y-.2)^2\right)\right)}$$

which has been studied extensively in Gu, Bates, Chen and Wahba (1989), Breiman (1991), Friedman (1991), He and Shi (1996), Hansen, Kooperberg and Sardy (1996), and Koenker and Mizera (2002). The (x_i, y_i) covariates are generated from independent uniforms on $[0,1]^2$ and the response is generated by

$$z_i = f_0(x_i, y_i) + u_i \quad i = 1, \cdots, n \tag{13}$$

where u_i is generated as standard normal random variable. Notice that in the last two equations we have reverted temporarily back to the notations commonly used in the statistics literature. This bivariate surface is then fitted with the penalized triogram introduced in Koenker and Mizera (2002) which is formulated as the following regression quantile problem:

$$\min_{b \in R^n} \sum_{i=1}^{n} \rho(z_i - g_i' b) + \lambda \sum_{k=1}^{M} |h_k' b| \tag{14}$$

where g_i is the pseudo design vector with elements $g_{ij} = (B_j(x_i, y_i))$, B_j is a barycentric basis function, h_k is the roughness vector that reflects the change in gradients along the M edges of the triogram and λ is the smoothing parameter that controls the trade-off between fidelity to the data measured in the first term and roughness of the fit captured in the second summation. To express (14) as the regression quantile problem in (1), we introduce the $(n+M)\times n$ pseudo design matrix $X = \left[G' \vdots H'\right]'$ where $G = (g_i')$, $H = (h_k')$, and the pseudo response vector $y = (z', 0') \in \mathbb{R}^{n+M}$. This can then be solved as the linear program in (3) or (4).

Reverting back to the numerical analysis notations, the structure of a typical pseudo design matrix A' in (4) of the triogram is presented in Figure 1. Figure 2 contains the pattern of the $AQ^{-1}A'$ matrix. It is apparent from Figure 2 that the matrix to be

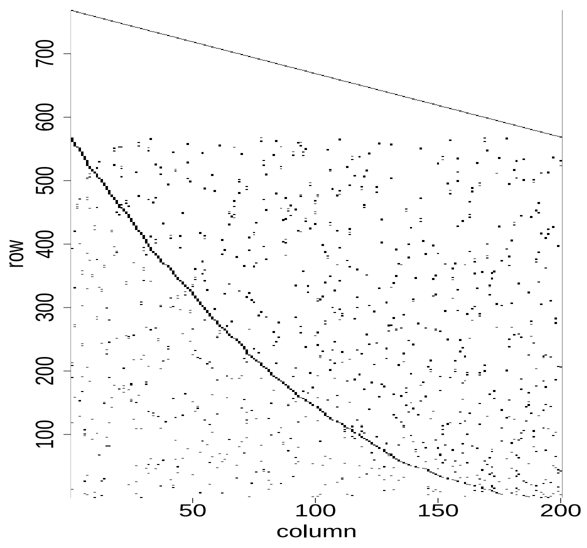

Figure 1. A typical *pseudo* design matrix of the penalized triogram problem in Koenker and Mizera (2002)

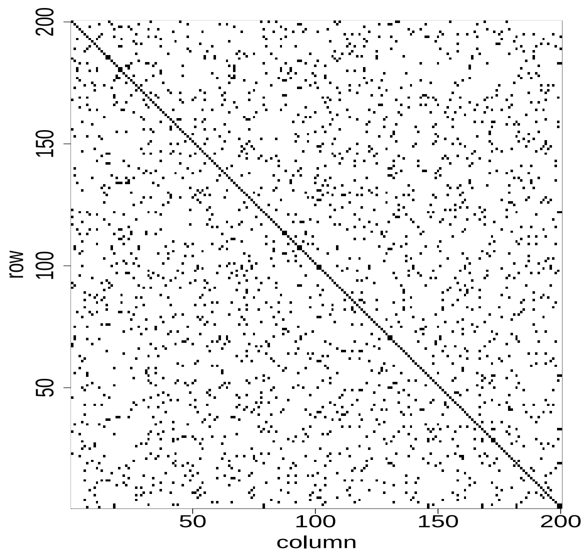

Figure 2. A typical $AQ^{-1}A'$ matrix of the penalized triogram problem in Koenker and Mizera (2002)

factorized is extremely sparse. The portion of nonzero entries in $AQ^{-1}A'$ is roughly $13/n$. Separate timing of the four different steps in Ng and Peyton's algorithm is presented in Figure 3. The amount of time reported is the median execution time of solving the linear system of equations with $AQ^{-1}A'$ as the coefficient matrix and the pseudo response vector in (14) as the right-hand-side in 50 replications of (13) for sample size ranges from 1000 to 50000. The timing grows exponentially. The rates of growth are reflected in the value of the least-squares slope coefficient b_1 of the log-linear model fitted to the data points in Figure 3. The biggest chunk of time is expended in numerical factorization followed by minimum degree ordering. The fact that the nonzero pattern of the $AQ^{-1}A'$ matrix remains unchanged from one iteration to the other enable us to perform the ordering step only once over the whole iteration process in the Frisch-Newton algorithm. With a twenty-iteration execution, for example, an additional saving of roughly a factor of 1/5 in computing time can further be realized.

The remaining linear algebra in the system of equations in (8) and (12) is performed with the routines available in SparseM, see Koenker and Ng (2002), which utilizes BLAS-like routines tailored for sparse matrices from SPARSKIT2.

6. PERFORMANCE

We perform a small scale simulation using the penalized triogram in (14) to fit the bivariate model in (13) to study the performance of our sparse implementation of the Frisch-Newton algorithm. We compare our sparse implementation (`srqfn`) to the implementation in Koenker and Mizera (2002), which utilizes the dense matrix version of the Frisch-Newton algorithm (`rqfn`) reported in Koenker and Portnoy (1997). Since it takes a long time for `rqfn` to solve the penalized triogram problem for just one replication for moderately large size, we only perform 10 replications in the simulation study for $n = 2^{(6 \text{ to } 10 \text{ by } 0.5)}$. Both implementations of the Frisch-Newton algorithm are coded in FORTRAN while the interfaces are in **R**. The simulation is performed on a Sun Sparcstation and the timings are clocked between the time the computation enters and leaves the Fortran code that performs the Frisch-Newton iterations, hence, eliminating any discrepancy on the overhead between the two implementation.

Figure 4 reports the median execution time required to compute the triogram solutions. The advantage of `srqfn` over `rqfn` is prominent in Figure 4 over the whole range of sample sizes we have investigated. The factor of improvement ranges from roughly 36 at the sample size of 64 to approximately 852 at the sample size of 1024. Also reported in the legend of Figure 4 are the least-squares estimated coefficients of the log-linear model fitted to execution time on sample size. In Figure 5, we report the timing for only the `srqfn` for $n = 2^{(6 \text{ to } 14 \text{ by } 0.5)}$. Also included in the figure are the least-squares estimated intercept and slope coefficients from regressing the log of execution time on the log of sample size. The estimated slope coefficient is almost

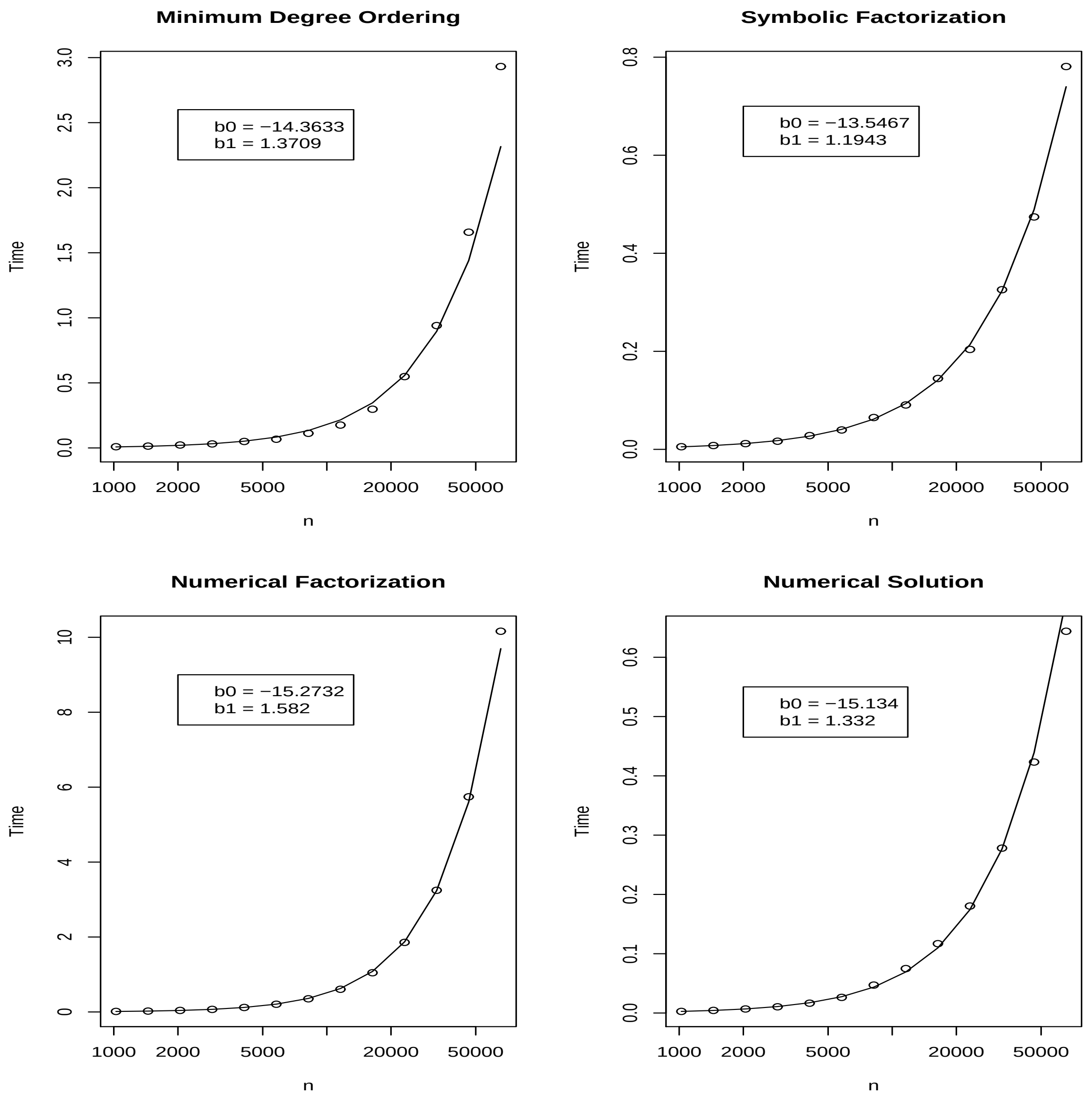

Figure 3. Median timing in seconds for the steps in Ng and Peyton's (1999) algorithm of one least-squares solution over 50 replications as a function of sample size

identical to that in Figure 4, which suggests execution time grows exponentially at the rate of around 1.54.

To compare the storage saving from our sparse implementation, we report in Figure 6 the storage size, in bytes, needed to compute the penalized triogram solutions for different sample sizes on our Sun Sparcstation. What is evident from the figure is the

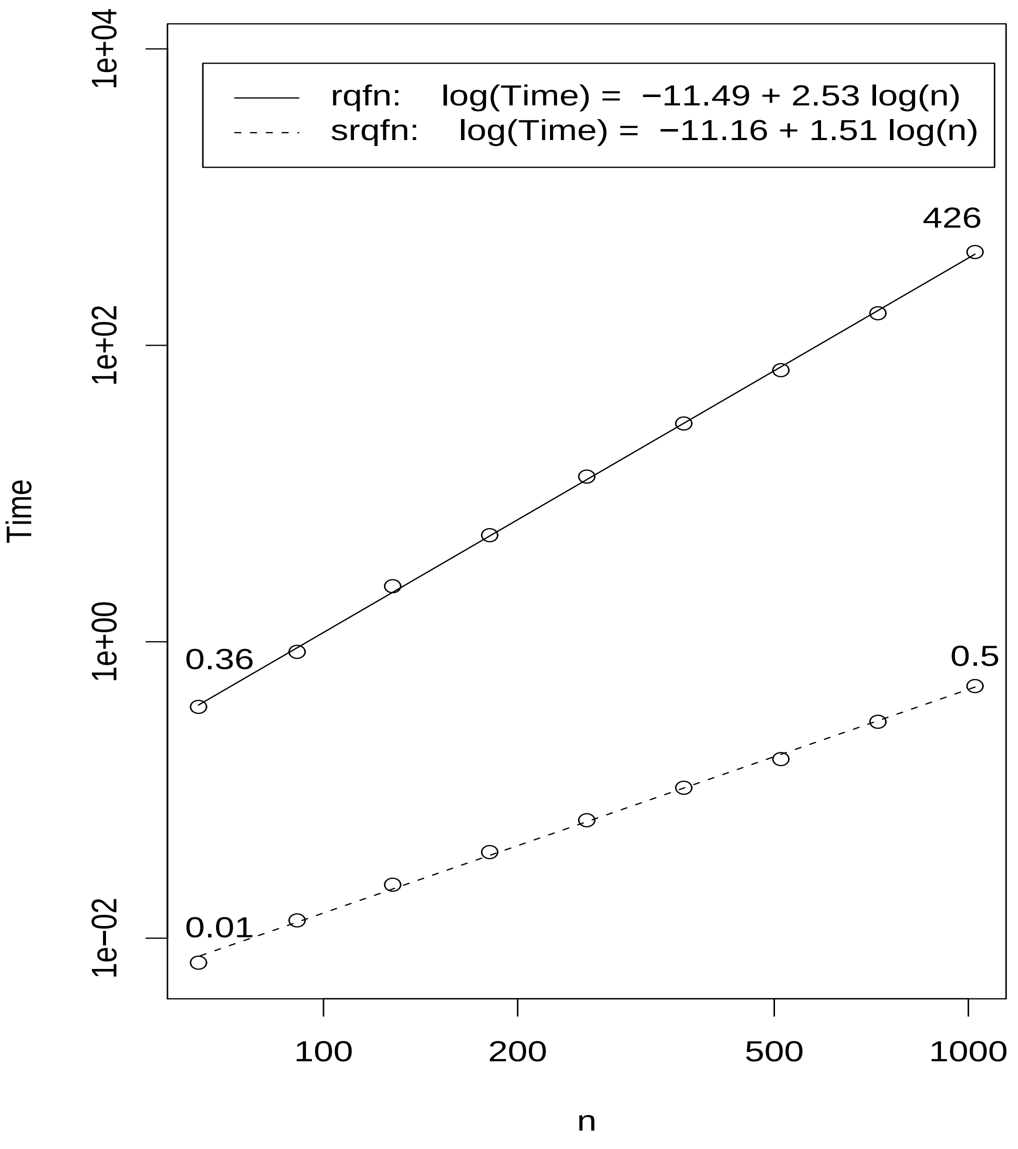

FIGURE 4. Median execuation time to obtain the penalized triogram solution to model (13) for `rqfn` and `srqfn`.

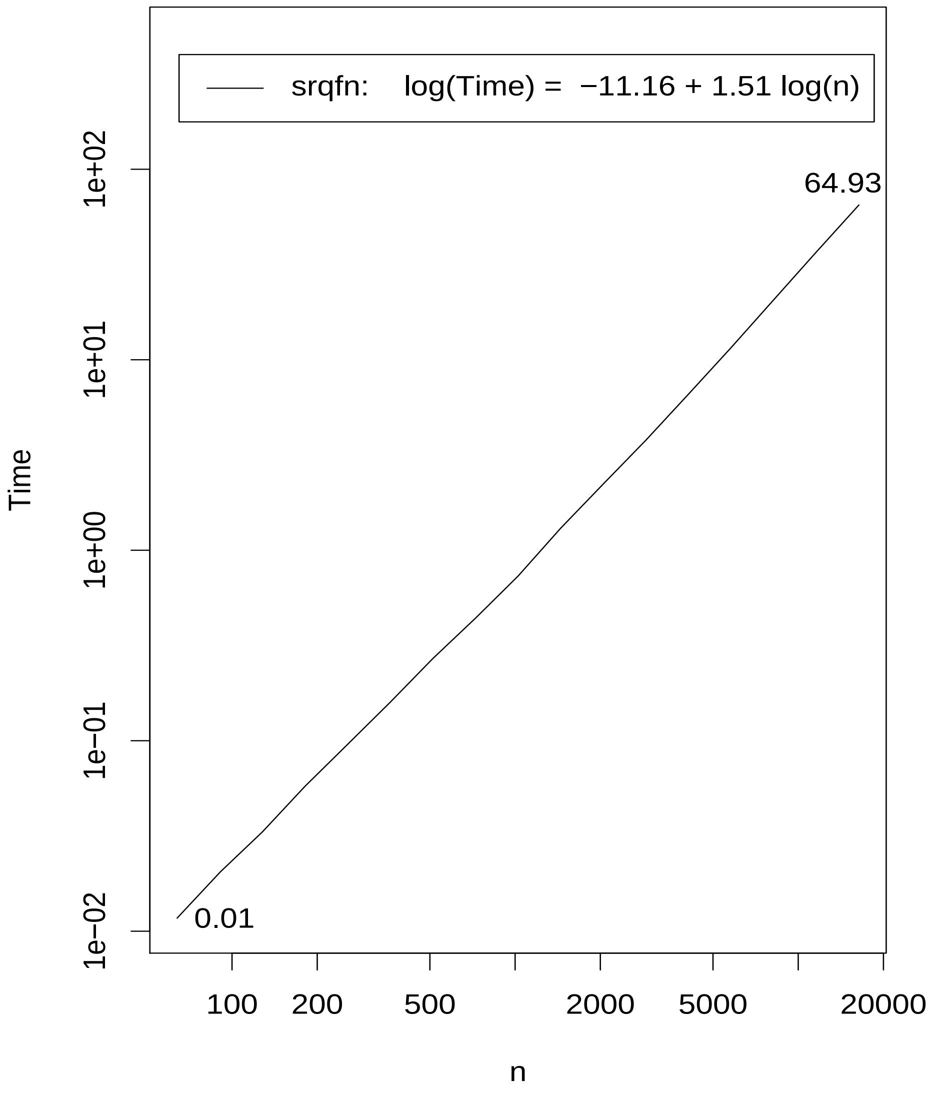

Figure 5. Median execution time of srqfn.

exponentially increasing storage requirement of `rqfn` compared to the linear increase in `srqfn`.

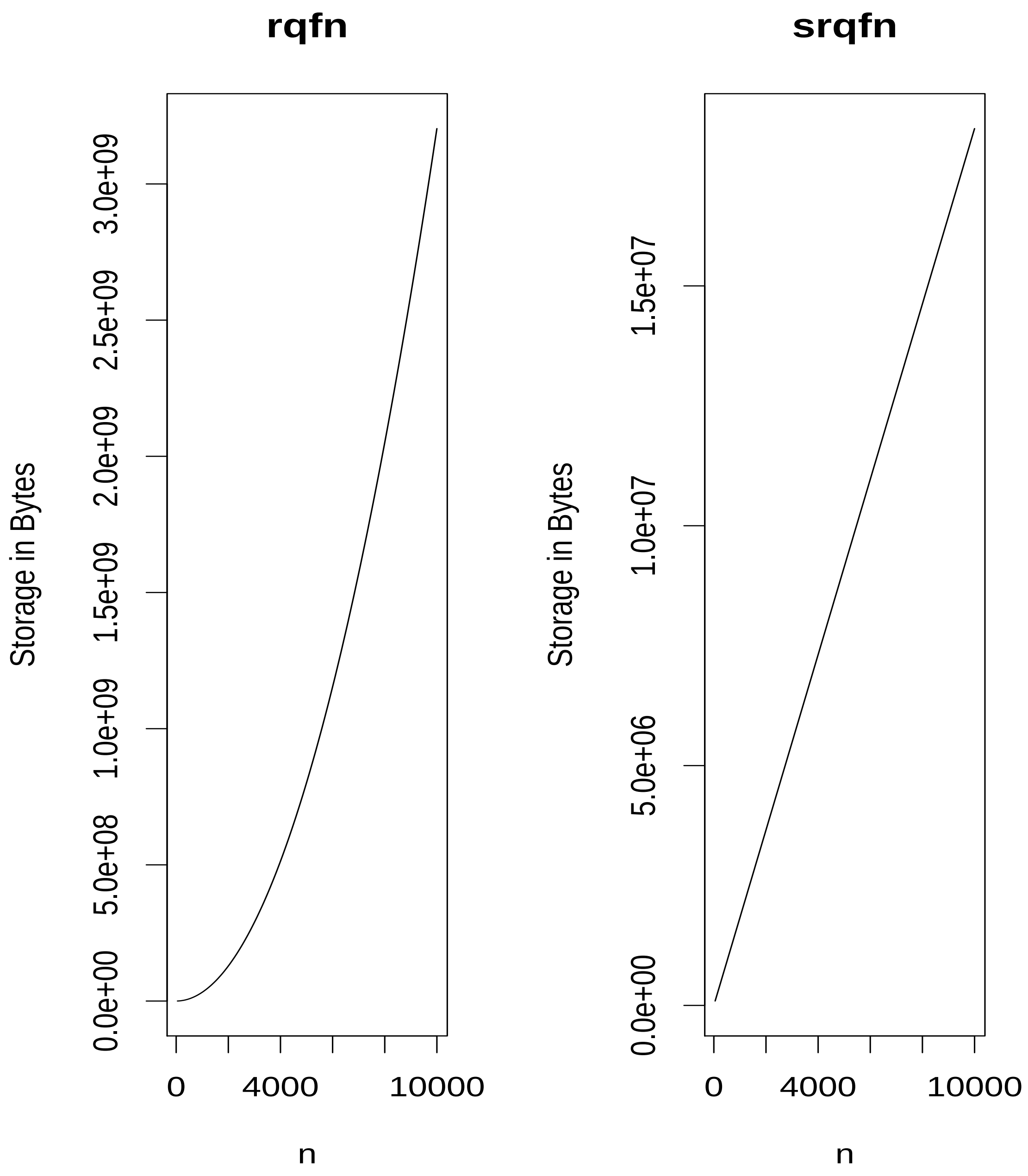

FIGURE 6. Storage requirement in bytes for the rqfn and srqfn implementation when used to estimate the penalized triogram model.

References

Breiman, L., (1991). "The Π method for estimating multivariate functions from noisy data (Disc: 145-160)", *Technometrics*, 33, 125-143.

Friedman, J.H., (1991). "Multivariate adaptive regression splines (Disc: 67-141)", *The Annals of Statistics*, 19, 1-67.

Frisch, K.R., (1955). "*The logarithmic potential method of convex programming*", Technical Report, University Institute of Economics, Oslo, Norway.

Gonzaga, C., (1992). "Path-following methods in linear programming", *SIAM Review*, 34, 167-224.

Gu, C., D.M. Bates, Z. Chen, and G. Wahba, (1989). "The computation of generalized cross-validation functions through Householder tridiagonalization with applications to the fitting of interaction spline models", *SIAM Journal on Matrix Analysis and Applications*, 10, 457-480.

Hansen, M., C. Kooperberg, and S. Sardy, (1998). "Triogram models", *Journal of the American Statistical Association*, 93, 101-119.

He, X. and P. Shi, (1996). "Bivariate tensor-product B-splines in a partly linear model", *Journal of Multivariate Analysis*, 58, 162-181.

Koenker, R. and I. Mizera, (2002). "Penalized triograms: Total variation regularization for bivariate smoothing", working paper.

Koenker, R. and P. Ng, (2002). "SparseM: a sparse matrix package for R", working paper.

Koenker, R. and S. Portnoy (1997). "The Gaussian hare and the Laplacian tortoise: computability of squared-error vs absolute error estimators", (with discussions), *Statistical Science*, 12, 279-300.

Liu, J. W-H., (1985). "Modification of the minimum degree algorithm by multiple elimination", *ACM Trans. Math. Software*, 11, 141-153.

Lustig, I.J., R.E. Marsten and D.F. Shanno, (1994). "Interior point methods for linear programming: Computational state of the art", *ORSA Journal on Computing*, 6, 1-14.

Ng, E.G. and B.W. Peyton, (1993) "Block sparse Cholesky algorithms on advanced uniprocessor computers", *SIAM J. Sci. Comput.*, Volume 14, pp. 1034-1056.

Saad, Y., (1994). Sparskit: A basic tool kit for sparse matrix computations; Version 2, available from: `www.cs.umn.edu/Research/arpa/SPARSKIT/sparskit.html`

Wright, S., (1997). *Primal-Dual Interior-Point Methods*, Society for Industrial and Applied Mathematics, Philadelphia.

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

NORTHERN ARIZONA UNIVERSITY