THE DANTZIG DECODER RING: AN R VINAIGRETTE

ROGER KOENKER

Abstract. An attempt is made to demystify the Dantzig decoding procedure of Candes and Randall (2008).

1. Introduction

As a secret agent of the deep state, your job is to transmit messages back to your superiors who are in exile somewhere in wilderness of Idaho. Messages consist of a vector of real numbers, $x \in \mathbb{R}^p$. Transmission is over a noisy channel so the following protocol has been developed. You and your superiors share knowledge of a n by p matrix A with $n \gg p$. You send the enlarged message $y = Ax$ through the noisy channel. On good days they receive the message $\hat{y} = y + u$ where $u \in \mathbb{R}^n$ is a n-vector with iid (standard) Gaussian components. On such good days decoding the messages is relatively simple: the receiver can just compute the least squares projection of $\hat{y}$ on the column space of A,

$$\hat{x} = (A^\top A)^{-1} A^\top \hat{y},$$

and the message will be denoised relatively accurately, depending to some degree on the choice of n and A.

However, on bad days the noise in transmission can be much worse. In particular, it may contain a limited number of very large errors that can severely disrupt the quality of the $\hat{x}$ solution. More explicitly, suppose that on bad days, instead of $\hat{y}$ the vector $\tilde{y} = y + u + v$ is received with v being arbitrary, but relatively sparse in the sense that most of its coordinates are zero. Decoding under these conditions is considerably more difficult, but you have developed the following alternative protocol. Let,

$$\tilde{x} = (A^\top A)^{-1} A^\top (\tilde{y} - \tilde{v}),$$

where

$$\tilde{v} = \operatorname{argmin}\{\|v\|_1 \text{ such that } \|Q(\tilde{y} - v)\|_\infty \leq K\}$$

and $Q = I - A(A^\top A)^{-1} A^\top$. This form of denoising is closely related to the so-called Dantzig selector of Candes and Tao (2007), which we will now describe in more detail.

July 20, 2024. An earlier version this note was written in 2007 for an exercise in my Econometric Theory course. An extensive collection of references and code for related methods can be found at Candes (2007). A genre manifesto for R Vinaigrettes is available at `http://davoidofmeaning.blogspot.com/2016/12/r-vinaigrettes.html`.

2. The Dantzig Selector

The basic idea of the Dantzig selector is to solve regression problems that look like this:

$$\min\{\|b\|_1 \mid \|X^\top r\|_\infty < K\}.$$

where $r = y - Xb$. This looks a lot like the familiar lasso penalized least squares regression problem, and that's because it is very similar. Rather than adding an ℓ_1 penalty to the usual least squares regression objective we minimize the ℓ_1 norm of the coefficients subject to a ℓ_∞ norm constraint on the gradient of the least squares criterion. Rather than solving a quadratic programming problem we have a somewhat simpler linear programming problem instead.

We consider a slightly more general version:

$$\min\{\|b\|_1 \mid K \leq X^\top r \leq K\}.$$

for vector valued K. We can implement this version with the inequality constrained quantile regression function, `rq.fit.fnc(X, y, R, r)` from the R package **quantreg**. This function solves problems of the form:

$$\min\{\|y - Xb\|_1 \mid Rb >= r\}$$

If we set, $X = I_p$ the p dimensional identity matrix and $y = 0 \in \mathbb{R}^p$ with $R = [X^\top X \vdots - X^\top X]$ and $r = [(X^\top y - K)^\top, [(-X^\top y - K)^\top]^\top$ we have the Dantzig selector as a special case. Obviously, the parameter K controls the amount of shrinkage of the coefficients b.

3. Dantzig Decoding

Returning to our denoising problem, let V denote the subspace spanned by the columns of A and $V^\perp$ the orthocomplement of V. We first construct a matrix, Q, whose columns form an orthobasis for $V^\perp$, so $Q^\top A = 0$. Now, we can solve for $\tilde{v}$ with the Dantzig selector, setting $X = Q^\top$ and $y = Q^\top \tilde{y}$. It only remains to choose the tuning parameter K. This can be somewhat delicate, since it depends upon the scale of the Gaussian noise vector, u, and other features of the problem. We adopt a simple expedient in the illustrative code below. Actually, this isn't quite the end of it. Like the well-known feature/bug of the lasso, the Dantzig selector also tends to act too agressively on some coordinates, so Candes and Randall (2008) recommend a "reprojection" step that adjusts for this. Code for both the Dantzig selector and the Dantzig Decoder in the R languange appears in the next section.

4. A Simulated Example

In this section we illustrate the use of the Dantzig decoder with a simple example. The original message consists of a $p = 256$ iid Gaussian vector. The shared matrix A is $n = 2p$ by p with iid Gaussian entries. A fraction $\rho = 0.1$ of the coordinates of the vector, $y = Ax$, to be transmitted are contaminated by replacing their intended values by the negative of those values. Finally, Gaussian noise is added to the transmitted vector.

In Figure 1 we show three scatter plots each plotting a decoded version of the message against its true values, x. In the left panel we plot $\hat{x}$ using the naive least squares decoding. In the middle panel we plot the Dantzig decoding, $\tilde{x}$. And in the right panel we plot an oracle decoding achievable if the contamination component v were known. In each panel we report root mean squared error of the respective decodings. It can be seen that the Dantzig decoding is almost as good as the oracle decoding, while the Gaussian decoding is roughly 10 times noisier than the Dantzig method.

```
DantzigSelector <- function(X,y,K){
    require(quantreg)
    n <- nrow(X)
    p <- ncol(X)
    A <- crossprod(X)
    R <- rbind(A, -A)
    a <- crossprod(X,y)
    r <- c(a-K, -a-K)
    zp <- rep(0,p)
    Ip <- diag(p)
    f <- rq.fit.fnc(Ip, zp, R=R, r=r)
    return(f)
}
DantzigDecoder <- function(A,y,sigma,Kval = 3){
   NullSpace <- function(A){
        S <- svd(A,nrow(A))
        r <- length(S$d)
        S$u[,(r+1):nrow(A),drop=FALSE]
       }
   Q <- t(NullSpace(A))
   Qy <- Q %*% y
   K <- sqrt(diag(crossprod(Q))) * sigma * Kval
   f <- DantzigSelector(Q, Qy, K)
   # Reprojection Step
   s <- abs(f$coef) > sigma
   e <- rep(0,nrow(A))
   e[s] <- lm(Qy ~ Q[,s] -1)$coef
   xhat <- lm((y-e) ~ A - 1)$coef
   return(xhat)
  }
```

```
p <- 256
n <- 2 * p
set.seed(1729)
A <- matrix(rnorm(p*n),n,p)
rho <- 0.10
k <- round(rho * n)
x <- rnorm(p)
y <- A %*% x
yt <- y
sigma <- median(abs(y))/16
s <- sample(1:n,k)
yt[s] <- -yt[s] # Original Candes example
u <- rnorm(n)*sigma
yt <- yt + u
xhat <- DantzigDecoder(A, yt, sigma)
```

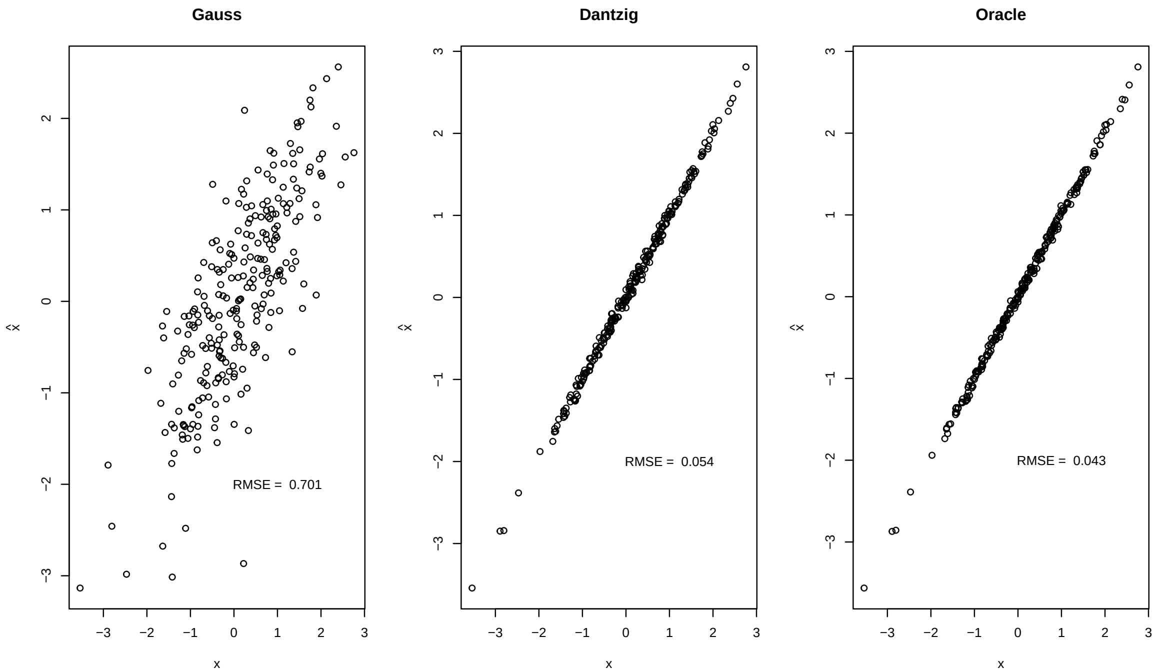

Figure 1. RMSE Effectiveness of Gaussian and Dantzig Decoding

```
par(mfrow=c(1,3))
g <- coef(lm(yt ~ A -1))
rmseG <- round(sqrt(var(x-g)),3)
plot(x,g,ylab=expression(hat(x)),main="Gauss")
text(1,-2,paste("RMSE = ",rmseG))
plot(x,xhat,ylab=expression(hat(x)),main="Dantzig")
rmseD <- round(sqrt(var(x-xhat)),3)
text(1,-2,paste("RMSE = ",rmseD))
h <- coef(lm((y + u) ~ A - 1))
plot(x,h,ylab=expression(hat(x)),main="Oracle")
rmseH <- round(sqrt(var(x-h)),3)
text(1,-2,paste("RMSE = ",rmseH))
```

5. Conclusion

"OK, so it seems to work, but why?" To answer this question it is necessary to delve a bit deeper into the inner workings of median regression estimator. A hint about this is provided by the sparseness of the contamination vector, v. The Dantzig decoder acts like the connventional ℓ_1 regression estimator. As long as there are only a few non-zero entries in v, it is capable of easily identifying them, and once they are removed decoding is essentially as good as the oracle operating on purely Gaussian data.

References

Candes, E. (2007), 'ℓ_1 magic'. `https://candes.su.domains/software/l1magic/`.

Candes, E. J. and Randall, P. A. (2008), 'Highly robust error correction by convex programming', *IEEE Trans. Inf. Theor.* **54**, 2829 – 2840.

Candes, E. and Tao, T. (2007), 'The Dantzig selector: Statistical estimation when p is much larger than n', *The Annals of Statistics* **35**, 2313 – 2351.