DERIVATIVES OF THE PROCRUSTUS TRANSFORMATION WITH APPLICATIONS

JAN DE LEEUW AND KEKONA KAI SORENSON

ABSTRACT. We derive expressions for the derivatives of the Procrustus Transformation. As byproducts we find the derivatives of singular values and vectors and of the direct and inverse symmetric square root. The results are applied to minimization algorithms over Stiefel manifolds, and in particular to Polynomial Component and Factor Analysis.

1. PROCRUSTUS TRANFORMATION

Suppose X is an $n \times m$ matrix, with $n \geq m$ and $\textbf{rank}(X) = m$. The *Procrustus Transformation* $\Pi(X)$ is the $n \times m$ orthonormal matrix that minimizes the sum-of-squares $\textbf{SSQ}(Y - X)$, or, equivalently, the $n \times m$ orthonormal matrix that maximizes $\textbf{tr}\ Y'X$. See Cliff [1966].

The stationary equation for this optimization problem are

$$X = YM, \tag{1a}$$

$$Y'Y = I, \tag{1b}$$

with M a symmetric matrix of Lagrange Multipliers. It follows that $M^2 = X'X$, i.e. $M = (X'X)^{\frac{1}{2}}$, the unique symmetric square root of $X'X$. And thus $\Pi(X) = X(X'X)^{-\frac{1}{2}}$.

Date: Saturday 14th April, 2012 — 15h 9min — Typeset in LUCIDA BRIGHT.

Key words and phrases. Matrix derivatives, Stiefel manifolds, majorization algorithms, polynomial factor analysis.

It is common to define the Procrustus Tranformation using the *Singular Value Decomposition*. We write the SVD as $X = K\Lambda L'$, where the $n \times m$ matrix K satisfies $K'K = I$, the $m \times m$ matrix L satisfies $L'L = LL' = I$, and the diagonal matrix Λ has positive elements in non-increasing order along the diagonal. Then $X'X = L\Lambda^2 L'$, and $(X'X)^{\frac{1}{2}} = L\Lambda L'$, and $(X'X)^{-\frac{1}{2}} = L\Lambda L'$. Thus $\Pi(X) = KL'$.

2. Using the SVD

The singular value decomposition is defined as the solution to the equations

$$XL = K\Lambda, \tag{2a}$$

$$X'K = L\Lambda, \tag{2b}$$

$$K'K = I, \tag{2c}$$

$$L'L = I. \tag{2d}$$

Suppose the elements x_{ij} of X are differentiable functions of a single real variable, say θ. We use notation such as $\mathcal{D}X$ for matrices of derivatives with respect to θ. Thus $\{\mathcal{D}X)\}_{ij} = \mathcal{D}x_{ij}$ is an $n \times m$ matrix of functions, just like X.

Differentiating (2) gives

$$(\mathcal{D}X)L + X(\mathcal{D}L) = K(\mathcal{D}\Lambda) + (\mathcal{D}K)\Lambda, \tag{3a}$$

$$(\mathcal{D}X')K + X'(\mathcal{D}K) = L(\mathcal{D}\Lambda) + (\mathcal{D}L)\Lambda, \tag{3b}$$

$$(\mathcal{D}K)'K + K'(\mathcal{D}K) = 0, \tag{3c}$$

$$(\mathcal{D}L)'L + L'(\mathcal{D}L) = 0. \tag{3d}$$

Let $A \triangleq K'(\mathcal{D}X)L$. It follows immediately that $\mathcal{D}\Lambda = \mathbf{diag}(A)$. Also define $\overline{A} \triangleq A - \mathbf{diag}(A)$. Then

$$\overline{A} = K'(\mathcal{D}K)\Lambda - \Lambda L'(\mathcal{D}L),$$

$$\overline{A}' = L'(\mathcal{D}L)\Lambda - \Lambda K'(\mathcal{D}K).$$

Thus

$$\Lambda\overline{A} = \Lambda K'(\mathcal{D}K)\Lambda - \Lambda^2 L'(\mathcal{D}L),$$
$$\overline{A}'\Lambda = L'(\mathcal{D}L)\Lambda^2 - \Lambda K'(\mathcal{D}K)\Lambda,$$

and thus

$$\Lambda\overline{A} + \overline{A}'\Lambda = L'(\mathcal{D}L)\Lambda^2 - \Lambda^2 L'(\mathcal{D}L). \tag{4a}$$

Similarly

$$\overline{A}\Lambda = K'(\mathcal{D}K)\Lambda^2 - \Lambda L'(\mathcal{D}L)\Lambda,$$
$$\Lambda\overline{A}' = \Lambda L'(\mathcal{D}L)\Lambda - \Lambda^2 K'(\mathcal{D}K),$$

and thus

$$\overline{A}\Lambda + \Lambda\overline{A}' = K'(\mathcal{D}K)\Lambda^2 - \Lambda^2 K'(\mathcal{D}K). \tag{4b}$$

Suppose all singular values are different. Define the $m \times m$ anti-symmetric matrices E and F with zero diagonal elements and off-diagonal elements

$$e_{j\ell} \triangleq \frac{\lambda_j a_{j\ell} + \lambda_\ell a_{\ell j}}{\lambda_\ell^2 - \lambda_j^2}, \tag{5a}$$

$$f_{j\ell} \triangleq \frac{\lambda_\ell a_{j\ell} + \lambda_j a_{\ell j}}{\lambda_\ell^2 - \lambda_j^2}. \tag{5b}$$

From (4a) $L'(\mathcal{D}L) = E$ and thus $\mathcal{D}L = LE$. From (4b) $K'(\mathcal{D}K) = F$. Choose an $n \times (n-m)$ matrix K_0 satisfying $K'K_0 = 0$ and $K_0'K_0 = I$. Also define $A_0 \triangleq K_0'(\mathcal{D}X)L$. From (3a) we have $A_0 = K_0'(\mathcal{D}K)\Lambda$, which means $\mathcal{D}K = KF + K_0F_0$, where $F_0 \triangleq A_0\Lambda^{-1}$. To write our result more compactly we define $\vec{K} \triangleq \begin{bmatrix} K & | & K_0 \end{bmatrix}$ and $\vec{F} \triangleq \begin{bmatrix} F \\ F_0 \end{bmatrix}$.

Lemma 2.1. *For the singular value decomposition $X = K\Lambda L'$ of an $n \times m$ matrix X, with $n \geq m$, we find*

$$\mathcal{D}\Lambda = \mathbf{diag}(A), \tag{6a}$$

$$\mathcal{D}L = LE, \tag{6b}$$

$$\mathcal{D}K = \vec{K}\vec{F}. \tag{6c}$$

Theorem 2.2.

$$\mathcal{D}\Pi = K(F-E)L' + K_0F_0L' = \overrightarrow{K}\overrightarrow{F}L' - KEL' \tag{7}$$

Proof. Use $\mathcal{D}\Pi = K(\mathcal{D}L') + (\mathcal{D}K)L'$ and apply the lemma. □

Note that for $i \neq j$

$$f_{j\ell} - e_{j\ell} = \frac{a_{j\ell} - a_{\ell j}}{\lambda_j + \lambda_\ell}, \tag{8}$$

which is again anti-symmetric.

3. Using the Symmetric Square Root

For archival purposes we give an alternative derivation that uses the definition $\Pi(X) = X(X'X)^{-\frac{1}{2}}$. Differentiating gives

$$\mathcal{D}\Pi = (\mathcal{D}X)(X'X)^{-\frac{1}{2}} + X\mathcal{D}((X'X)^{-\frac{1}{2}}). \tag{9}$$

Now

$$\mathcal{D}((X'X)^{-\frac{1}{2}}) = -(X'X)^{-\frac{1}{2}}\mathcal{D}((X'X)^{\frac{1}{2}})(X'X)^{-\frac{1}{2}}. \tag{10}$$

We also differentiate

$$(X'X)^{\frac{1}{2}}(X'X)^{\frac{1}{2}} = X'X, \tag{11}$$

and find

$$\mathcal{D}((X'X)^{\frac{1}{2}})(X'X)^{\frac{1}{2}} + (X'X)^{\frac{1}{2}}\mathcal{D}((X'X)^{\frac{1}{2}}) = (\mathcal{D}X)'X + X'(\mathcal{D}X). \tag{12}$$

Now $(X'X)^{\frac{1}{2}} = L\Lambda L'$. Define $H \triangleq L'\mathcal{D}((X'X)^{\frac{1}{2}})L$. Then

$$H\Lambda + \Lambda H = A'\Lambda + \Lambda A, \tag{13}$$

and thus

$$h_{j\ell} = \frac{\lambda_j a_{j\ell} + \lambda_\ell a_{\ell j}}{\lambda_j + \lambda_\ell}. \tag{14}$$

We see that $\mathcal{D}((X'X)^{\frac{1}{2}}) = LHL'$ and $\mathcal{D}((X'X)^{-\frac{1}{2}}) = -L\Lambda^{-1}H\Lambda^{-1}L'$.

Lemma 3.1.

$$\mathcal{D}((X'X)^{\frac{1}{2}}) = LHL', \tag{15a}$$

$$\mathcal{D}((X'X)^{-\frac{1}{2}}) = -L\Lambda^{-1}H\Lambda^{-1}L', \tag{15b}$$

where H is given by (14)*.*

Theorem 3.2.

$$\mathcal{D}\Pi = (\mathcal{D}X)(X'X)^{-\frac{1}{2}} - KH\Lambda^{-1}L'. \tag{16}$$

Proof. □

For both (7) and (16) we see that $K_0'(\mathcal{D}\Pi)L = K_0'(\mathcal{D}X)L\Lambda^{-1}$ and $K'(\mathcal{D}\Pi)L = F - E = (A - H)\Lambda^{-1}$. Thus the two expressions for $\mathcal{D}\Pi$ indeed give identical results.

4. Partial Derivatives

$$\frac{\partial\{\Pi\}_{is}}{\partial x_{kt}} = e_i'\{e_k e_t'(X'X)^{-\frac{1}{2}} - KH\Lambda^{-1}L'\}e_s,$$

where

$$h_{st} = \frac{\lambda_s a_{st} + \lambda_t a_{ts}}{\lambda_s + \lambda_t}.$$

5. Minimization over the Orthonormals

Consider the constrained minimization problem of minimizing $f(X)$ over the $n \times m$ matrices X satisfying $X'X = I$. This is equivalent to the unconstrained problem of minimizing $g(X) \stackrel{\Delta}{=} f(\Pi(X))$ over all $n \times m$ matrices X. By the chain rule the derivative is

$$\mathcal{D}g(X) = \mathcal{D}f(\Pi(X))\mathcal{D}\Pi(X),$$

which shows the relevance of our previous calculations.

Consider the Procrustus example of maximizing $f(X) = \mathbf{tr}\ Z'X$ over $X'X = I$. Instead we maximize $g(X) = \mathbf{tr}\ Z'\Pi(X)$ over all X.

Here is some simple R code using the optimx package [Nash and Varadhan, 2011].

```
require ("optimx")

set.seed (12345)
z <- matrix (rnorm (36), 9, 4)
x <- matrix (sample (1 : 36), 9, 4)

crussy <- function (x, z) {
    x <- matrix (x, nrow (z), ncol (z))
    s <- svd (x)
    p <- tcrossprod (s $ u, s $ v)
    return (sum (p * z))
}

res <- optimx (par = as.vector (x), fn = crussy, method = "
   BFGS", control = list (trace = 1, maximize = TRUE), z =
   z)
```

The code uses numerical differentiation, so it does not need a function to compute derivates. We can add a routine to compute the derivatives.

```
grussy <- function (x, z) {
    n <- nrow (z)
    m <- ncol (z)
    x <- matrix (x, n, m)
    g <- matrix (0, n, m)
    s <- svd (x)
    k <- s $ u
    l <- s $ v
    d <- s $ d
    q <- l %*% (t (l) / d)
    for (i in 1 : n) {
        for (j in 1 : m) {
            o <- outer (k [i, ], l[j, ])
            h <- (d * o + t (d * o)) / outer (d, d, "+")
            u <- k %*% h %*% (t (l) / d)
```

```
16              v <- outer (ifelse (i == (1 : n), 1, 0), g[j,
                    ]) - u
17              g [i, j] <- sum (z * v)
18          }
19      }
20      return (as.vector (g))
21  }
22
23  resg <- optimx (par = as.vector (x), fn = crussy, gr =
        grussy, method = c("BFGS"),
24      control = list (trace = 1, maximize = TRUE), z = z)
```

6. Application to Polynomial Component Analysis

$$f(X,B) = \mathbf{SSQ}(Y - \mathbf{pol}(\Pi(X))B).$$
$$g(X) = \min_B f(X,B) = f(X,B(X)).$$
$$\mathcal{D}g(X) = \mathcal{D}_1 f(X,B(X))$$

Minimize g over all X, without constraints.

Appendix A. Code

```
library(polynom)

procdev<-function(n, m, y = 0, p = 4, eps = 1e-6){
    set.seed(12345)
    nm <- n * m
    f <- rep (list(1 : p), n * m)
    for (i in 1 : nm) {
        f[[i]] <- polynomial (rnorm (p))
    }
    x <- matrix (0, n, m)
    g <- matrix (0, n, m)
    for (i in 1 : nm) {
        x[i] <- predict (f[[i]], y)
        g[i] <- predict (deriv (f[[i]]), y)
    }
    ss <- svd(x)
    sk <- ss$u
    sl <- ss$v
    sd <- ss$d
    px <- tcrossprod (sk, sl)
    sq <- tcrossprod (sl, (sl %*% diag (sd)))
    si <- tcrossprod (sl, (sl %*% diag (1/sd)))
    a <- crossprod (sk, g %*% sl)
    la <- sd * a
    e <- -(la + t(la)) / outer (sd ^ 2, sd ^ 2, "-")
    diag (e) <- 0
    la <- a %*% diag (sd)
    f <- -(la + t(la)) / outer (sd ^ 2, sd ^ 2, "-")
    diag (f) <- 0
    dl <- sl %*% e
    dk <- sk %*% f + (diag (n) - tcrossprod (sk)) %*% g %
        *% sl %*% diag (1/sd)
    dr <- tcrossprod(sk, dl) + tcrossprod (dk, sl)
    return (list (x=x, px=px, dr=dr))
}
```

References

N. Cliff. Orthogonal Rotation to Congruence. *Psychometrika*, 31: 33–42, 1966.

J.C. Nash and R. Varadhan. Unifying Optimization Algorithms to Aid Software System Users: optimx for R. *Journal of Statistical Software*, 43(9):1–13, 2011. URL http://www.jstatsoft.org/v43/i09/paper.

Department of Statistics, University of California, Los Angeles, CA 90095-1554

E-mail address, Jan de Leeuw: deleeuw@stat.ucla.edu

E-mail address, Kekona Kai Sorenson: kekona@gmail.com

URL, Jan de Leeuw: http://gifi.stat.ucla.edu

URL, Kekona Kai Sorenson: http://www.kekonakai.com/Site/Welcome.html