Solutions and Hints for Selected Exercises *Matrix Algebra* by James E. Gentle

Exercises Beginning on Page 63

2.1a. Assume $\{v_1, v_2, \ldots, v_k\}$ is a linearly independent set of vectors. Now consider the pair of vectors v_i and v_k. If $\{v_i, v_j\}$ is not linearly independent, then there exist a_i and a_j, not both 0, such that $a_i v_i + a_j v_j = 0$. In that case, however, we would have

$$a_1 v_1 + \cdots + a_i v_i + \cdots + a_j v_j + \cdots + a_k v_k = 0,$$

in which one $a_i, a_j \neq 0$, contradicting the condition that the full set is linearly independent.

2.1b. Let $v_1 = (1, 1, 0)$, $v_2 = (1, 0, 1)$, and $v_3 = (2, 1, 1)$. Now $\{v_1, v_2\}$, $\{v_1, v_3\}$, and $\{v_2, v_3\}$ are all linearly independent sets. We can see this because each pair has a vector with a zero that is unmatched by the other vector. The full set, however, is not linearly independent because $v_1 + v_2 - v_3 = 0$.

2.4b. Let one vector space consist of all vectors of the form $(a, 0)$ and the other consist of all vectors of the form $(0, b)$. The vector (a, b) is not in the union if $a \neq 0$ and $b \neq 0$.

2.9. Give a counterexample to the triangle inequality; for example, let $x = (9, 25)$ and $y = (16, 144)$.

2.12a. We first observe that if $\|x\|_p = 0$ or $\|y\|_q = 0$, we have $x = 0$ or $y = 0$, and so the inequality is satisfied because both sides are 0; hence, we need only consider the case $\|x\|_p > 0$ and $\|y\|_q > 0$. We also observe that if $p = 1$ or $q = 1$, we have the Manhattan and Chebyshev norms and the inequality is satisfied; hence we need only consider the case $1 < p < \infty$. Now, for p and q as given, for any numbers a_i and b_i, there are numbers s_i and t_i such that $|a_i| = \mathrm{e}^{s_i/p}$ and $|b_i| = \mathrm{e}^{t_i/q}$. Because e^x is a convex function, we have $\mathrm{e}^{s_i/p + t_i/q} \leq \frac{1}{p}\mathrm{e}_i^s + \frac{1}{q}\mathrm{e}_i^t$, or

$$a_i b_i \leq |a_i||b_i| \leq |a_i|^p/p + |b_i|^q/q.$$

Now let

$$a_i = \frac{x_i}{\|x\|_p} \quad \text{and} \quad b_i = \frac{y_i}{\|y\|_q},$$

and so

$$\frac{x_i}{\|x\|_p}\frac{y_i}{\|y\|_q} \leq \frac{1}{p}\frac{|x_i|^p}{\|x\|_p^p} + \frac{1}{q}\frac{|y_i|^q}{\|y\|_q^q}.$$

Now, summing these equations over i, we have

$$\begin{aligned}\frac{\langle x, y\rangle}{\|x\|_p\|y\|_q} &\leq \frac{1}{p}\frac{\|x\|_p^p}{\|x\|_p^p} + \frac{1}{q}\frac{\|y\|_q^q}{\|y\|_q^q} \\ &= 1.\end{aligned}$$

Hence, we have the desired result.
As we see from this proof, the inequality is actually a little stronger than stated. If we define u and v by $u_i = |x_i|$ and $v_i = |y_i|$, we have

$$\langle x, y\rangle \leq \langle u, v\rangle \leq \|x\|_p \|y\|_q.$$

We observe that equality occurs if and only if

$$\left(\frac{|x_i|}{\|x\|_p}\right)^{\frac{1}{q}} = \left(\frac{|y_i|}{\|y\|_q}\right)^{\frac{1}{p}}$$

and

$$\mathrm{sign}(x_i) = \mathrm{sign}(y_i)$$

for all i.
We note a special case by letting $y = 1$:

$$\bar{x} \leq \|x\|_p,$$

and with $p = 2$, we have a special case of the Cauchy-Schwarz inequality,

$$n\bar{x}^2 \leq \|x\|_2^2,$$

which guarantees that $\mathrm{V}(x) \geq 0$.

2.12b. Using the triangle inequality for the absolute value, we have $|x_i + y_i| \leq |x_i| + |y_i|$. This yields the result for $p = 1$ and $p = \infty$ (in the limit). Now assume $1 < p < \infty$. We have

$$\|x + y\|_p^p \leq \sum_{i=1}^n |x_i + y_i|^{p-1}|x_i| + \sum_{i=1}^n |x_i + y_i|^{p-1}|y_i|.$$

Now, letting $q = p/(p-1)$, we apply Hölder's inequality to each of the terms on the right:

$$\sum_{i=1}^n |x_i + y_i|^{p-1}|x_i| \leq \left(\sum_{i=1}^n |x_i + y_i|^{(p-1)q}\right)^{\frac{1}{q}} \left(\sum_{i=1}^n |x_i|^p\right)^{\frac{1}{p}}$$

and

$$\sum_{i=1}^n |x_i + y_i|^{p-1}|x_i| \leq \left(\sum_{i=1}^n |x_i + y_i|^{(p-1)q}\right)^{\frac{1}{q}} \left(\sum_{i=1}^n |y_i|^p\right)^{\frac{1}{p}},$$

so

$$\sum_{i=1}^n |x_i+y_i|^p \leq \left(\sum_{i=1}^n |x_i + y_i|^{(p-1)q}\right)^{\frac{1}{q}} \left(\left(\sum_{i=1}^n |x_i|^p\right)^{\frac{1}{p}} + \left(\sum_{i=1}^n |y_i|^p\right)^{\frac{1}{p}}\right)$$

or, because $(p-1)q = p$ and $1 - \frac{1}{q} = \frac{1}{p}$,

$$\left(\sum_{i=1}^n |x_i + y_i|^p\right)^{\frac{1}{p}} \leq \left(\sum_{i=1}^n |x_i|^p\right)^{\frac{1}{p}} + \left(\sum_{i=1}^n |y_i|^p\right)^{\frac{1}{p}},$$

which is the same as

$$\|x+y\|_p \leq \|x\|_p + \|y\|_p,$$

the triangle inequality.

2.21e. In $\mathbb{R}^3$,

$$\text{angle}(x,y) = \sin^{-1}\left(\frac{\|x \times y\|}{\|x\|\,\|y|}\right).$$

Because $x \times y = -y \times x$, this allows us to determine the angle from x to y; that is, the *direction* within $(-\pi, \pi]$ in which x would be rotated to y.

2.23. Just consider the orthogonal vectors $x = (1, 0)$ and $y = (0, 1)$. The centered vectors are $x_c = (\frac{1}{2}, -\frac{1}{2})$ and $y_c = (-\frac{1}{2}, \frac{1}{2})$. The angle between the uncentered vectors is $\pi/2$, while that between the centered vectors is π.

2.26a.

```
x <- c(1,2,3)
y <- c(-1,2,-3)
sum(x*y)
# [1] -6
```

2.26b.

```
Dot <- function(x, y){
  return(sum(x*y))
}
Dot(x, y)
# [1] -6
```

2.26c.

```
"%.%" <- function(x, y){
  return(sum(x*y))
}
x %.% y
# [1] -6
```

2.27.

```
x <- c(1,2,3)
y <- c(-1,2,-3)
c(x[2]*y[3]-x[3]*y[2],
  x[3]*y[1]-x[1]*y[3],
  x[1]*y[2]-x[2]*y[1])
# [1] -12   0   4
```

2.28.

```
x <- c(1,2,3)
sum(abs(x))
# [1] 6
sqrt(sum(x*x))
# [1] 3.741657
max(abs(x))
# [1] 3
```

2.29.

```
x <- c(1,2,3)
y <- x/sum(x*x)
y
# [1] 0.07142857 0.14285714 0.21428571
sum(x*y)
# [1] 1
```

2.30a.

```
x <- c(1,2,3)
y <- c(-1,2,-3)
ypx <- sum(x*y)*x/sum(x*x)
ypx
# [1] -0.4285714 -0.8571429 -1.2857143
```

2.30b.

```
c(x[2]*ypx[3]-x[3]*ypx[2],
  x[3]*ypx[1]-x[1]*ypx[3],
  x[1]*ypx[2]-x[2]*ypx[1])
# [1] -4.440892e-16  2.220446e-16  0.000000e+00
```

The actual cross product is $(0, 0, 0)$. In the solution above, two computed zeroes are of order -16, which is essentially an additive zero.

2.31a.

```
        x <- c(1,2,3)
        y <- c(-1,2,-3)
        z <- c(1,0,-3)
        u1 <- x; u2 <- y; u3 <- z
        u1 <- u1/sqrt(sum(u1*u1))
        u2 <- u2 - sum(u1*u2)*u1
        u3 <- u3 - sum(u1*u3)*u1
        u2 <- u2/sqrt(sum(u2*u2))
        u3 <- u3 - sum(u2*u3)*u2
        u3 <- u3/sqrt(sum(u3*u3))
```

2.31b.

```
        w <- c(5,4,3)
        c1 <-sum(w*u1)
        c2 <-sum(w*u2)
        c3 <-sum(w*u3)
        c1*u1 + c2*u2 + c3*u3
        # [1] 5 4 3
```

2.32.

```
        n <- 100
        set.seed(12345)
        u <- rnorm(n)
        z <- rnorm(n)
        v <- u+z
```

2.32a.

```
        mu <- mean(u)
        vu <- var(u)
        mv <- mean(v)
        vv <- var(v)
        round(c(mu,vu,mv,vv),2)
        # [1] 0.25 1.24 0.29 2.50
```

2.32b.

```
        covuv <- cov(u,v)
        coruv <- covuv/sqrt(vu*vv)
        round(c(covuv,coruv),2)
        # [1] 1.36 0.77
```

Exercises Beginning on Page 203

3.4. This exercise occurs in various guises in many different places, and the simple approach is to add and subtract $\bar{x}$:

$$\begin{aligned}(x-a)^{\mathrm{T}}(x-a) &= (x-\bar{x}-(a+\bar{x}))^{\mathrm{T}}(x-\bar{x}-(a+\bar{x}))\\ &= (x_{\mathrm{c}}-(a+\bar{x}))^{\mathrm{T}}(x_{\mathrm{c}}-(a+\bar{x}))\\ &= x_{\mathrm{c}}^{\mathrm{T}}x_{\mathrm{c}}+(a+\bar{x})^{\mathrm{T}}(a+\bar{x}))-2(a+\bar{x})^{\mathrm{T}}x_{\mathrm{c}}\\ &= x_{\mathrm{c}}^{\mathrm{T}}x_{\mathrm{c}}+n(a+\bar{x})^2.\end{aligned}$$

Finally, we get the expressions in equation (3.99) by writing $x_{\mathrm{c}}^{\mathrm{T}}x_{\mathrm{c}}$ as $\mathrm{tr}(x_{\mathrm{c}}x_{\mathrm{c}}^{\mathrm{T}})$.

3.16. Write the $n \times m$ matrix A as

$$A = (a_{ij}) = [a_{*1}, \ldots, a_{*m}].$$

If A is of rank one, the maximum number of linearly independent columns is one; hence, for $k = 2, \ldots, m$, $a_{*k} = c_k a_{*1}$, for some c_k.
Now let $x = a_{*1}$, which is an n-vector, and let y be an m-vector whose first element is 1 and whose $k = 2, \ldots, m$ elements are the c_ks. We see that $A = xy^{\mathrm{T}}$ by direct multiplication.
This decomposition is not unique, of course.

3.17. If the elements of the square matrix A are integers then each cofactor $a_{(ij)}$ is an integer, and hence the elements of $\mathrm{adj}(A)$ are integers. If $|A| = \pm 1$, then $A^{-1} = \pm\mathrm{adj}(A)$, and so the elements of A^{-1} are integers.
An easy way to form a matrix whose determinant is ± 1 is to form an upper triangular matrix with either 1 or -1 on the diagonal. A more "interesting matrix" that has the same determinant can then be formed by use of elementary operations. (People teaching matrix algebra find this useful!)

3.18. Because the inverse of a matrix is unique, we can verify each equation by multiplication by the inverse at the appropriate place. We can often reduce an expression to a desired form by multiplication by the identity MM^{-1}, where M is some appropriate matrix. For example, equation (3.187) can be verified by the equations

$$\begin{aligned}(A+B)(A^{-1}-A^{-1}(A^{-1}+B^{-1})^{-1}A^{-1}) &= \\ I-(A^{-1}+B^{-1})^{-1}A^{-1}+BA^{-1}-BA^{-1}(A^{-1}+B^{-1})^{-1}A^{-1} &= \\ I+BA^{-1}-(I+BA^{-1})(A^{-1}+B^{-1})^{-1}A^{-1} &= \\ (I+BA^{-1})(I-(A^{-1}+B^{-1})^{-1}A^{-1}) &= \\ B(B^{-1}+A^{-1})(I-(A^{-1}+B^{-1})^{-1}A^{-1}) &= \\ B(B^{-1}+A^{-1})-BA^{-1} &= I\end{aligned}$$

3.21. Express the nonzero elements of row i in the $n \times n$ matrix A as $a_i b_k$ for $k = i, \dots, n$, and the nonzero elements of column j as $a_k b_j$ for $k = j, \dots, n$. Then obtain expressions for the elements of A^{-1}. Show, for example, that the diagonal elements of A^{-1} are $(a_i b_i)^{-1}$.

3.25. For property 8, let c be a nonzero eigenvalue of AB. Then there exists v ($\neq 0$) such that $ABv = cv$, that is, $BABv = Bcv$. But this means $BAw = cw$, where $w = Bv \neq 0$ (because $ABv \neq 0$) and so c is an eigenvalue of BA. We use the same argument starting with an eigenvalue of BA. For square matrices, there are no other eigenvalues, so the set of eigenvalues is the same.

For property 9, see the discussion of similarity transformations on page 164.

3.37. Let A and B be such that AB is defined.

$$\begin{aligned}\|AB\|_{\mathrm{F}}^2 &= \sum_{ij}\left|\sum_k a_{ik}b_{kj}\right|^2 \\ &\leq \sum_{ij}\left(\sum_k a_{ik}^2\right)\left(\sum_k b_{kj}^2\right) \qquad \text{(Cauchy-Schwarz)} \\ &= \left(\sum_{i,k} a_{ik}^2\right)\left(\sum_{k,j} b_{kj}^2\right) \\ &= \|A\|_{\mathrm{F}}^2\|B\|_{\mathrm{F}}^2.\end{aligned}$$

3.40. Hints.

For inequality (3.325), use the Cauchy-Schwartz inequality.

For inequality (3.327), use Hölder's inequality.

3.45a.

```
Epqa <- function(n,p=0,q=0,a=0){
#  Function to create an elementary operator matrix.
#  The operator matrix may be E_pq, E_p(a), or E_pq(a).
    Epqa <- NA
```

```
            if (n<2){
                print('n must be at least 2')
                return()
            }
            perm <- FALSE
            mult <- FALSE
            axpy <- FALSE
            if (p>0&q>0&a==0) perm <- TRUE
            if (p>0&q==0) mult <- TRUE
            if (p>0&q>0&a!=0) axpy <- TRUE
            if (perm+mult+axpy==0){
                print('No valid elementary operator matrix')
                return()
            }
        # Set Epqa to the identity
            Epqa <- matrix(c(1,rep(c(rep(0,n),1),n-1)),ncol=n)
            if (perm){  # permute rows
                Epqa[p,p] <- 0
                Epqa[p,q] <- 1
                Epqa[q,q] <- 0
                Epqa[q,p] <- 1
            } else
            if (mult){  # multiply one row by a
                Epqa[p,p]=a
            } else
            if (axpy){  # axpy
                Epqa[p,q] <- a
            }
            return(Epqa)
        }
```

3.45b. E_{pq} is its own inverse.

```
        n <- 5
        p <- 2
        q <- 4
        E <- Epqa(n,p,q)
        E%*%E
```

3.45c. For $a \neq 0$, $E_p(1/a)E_p(a) = I$.

```
        n <- 5
        p <- 2
```

```
q <- 0
a <- 3
E1 <- Epqa(n,p,q,a)
E2 <- Epqa(n,p,q,1/a)
E1%*%E2
```

3.45d. An axpy operation, $E_{pq}(-a)E_{pq}(a) = I$.

```
n <- 5
p <- 2
q <- 4
a <- 3
E1 <- Epqa(n,p,q,a)
E2 <- Epqa(n,p,q,-a)
E1%*%E2
```

Exercises Beginning on Page 247

4.4. Let A is an $n \times n$ matrix, whose columns are the same as the vectors a_j, and let QR be the QR factorization of A. Because Q is orthogonal, $\det(Q) = 1$, and $\det(R) = \det(A)$. Hence, we have

$$\begin{aligned} |\det(A)| &= |\det(R)| \\ &= \prod_{j=1}^{n} |r_{jj}| \\ &\le \prod_{j=1}^{n} \|r_j\|_2 \\ &= \prod_{j=1}^{n} \|a_j\|_2, \end{aligned}$$

where r_j is the vector whose elements are the same as the elements in the j^{th} column of R.

If equality holds, then either some a_j is zero, or else $r_{jj} = \|r_j\|$ for $j = 1, \ldots, n$. In the latter case, R is diagonal, and hence $A^{\mathrm{T}}A$ is diagonal, and so the columns of A are orthogonal.

4.10. The R code that will produce the graph is

```
x<-c(0,1)
y<-c(0,1)
z<-matrix(c(0,0,1,1),nrow=2)
```

```
        trans<-persp(x, y, z, theta = 45, phi = 30)
        bottom<-c(.5,0,0,1)%*%trans
        top<-c(.5,1,1,1)%*%trans
        xends<-c(top[,1]/top[,4],bottom[,1]/bottom[,4])
        yends<-c(top[,2]/top[,4],bottom[,2]/bottom[,4])
        lines(xends,yends,lwd=2)
```

4.12a.

```
        Gaus <- function(n,p,ap,tol=sqrt(.Machine$double.eps)){
            if (tol*min(abs(ap[-p]))>abs(ap[p])) return(NA)
            Gaus <- Epqa(n,p,0,1/ap[p])
            if (p>1) {
            for (i in 1:(p-1)) Gaus <- Epqa(n,i,p,-ap[i])%*%Gaus
            }
            if (n>p) {
            for (i in (p+1):n) Gaus <- Epqa(n,i,p,-ap[i])%*%Gaus
            }
        return(Gaus)
        }
```

4.12b.

```
        n <- 3
        A <- matrix(c(1,3,2,2,3,1,2,1,2,4,2,1),nrow=n)
        p <- 2
        Gaus(n,p,A[,p])%*%A
```

4.12c.

```
        m <- dim(A)[2]
        A12 <- cbind(A,matrix(rep(0,n*n),nrow=n))
        for (i in 1:n) A12[i,m+i]<-1
        A12 <- Gaus(3,1,A12[,1])%*%A12
        A12 <- Gaus(3,2,A12[,2])%*%A12
        A12 <- Gaus(3,3,A12[,3])%*%A12
        A12 <- rbind(A12[,5:7],c(0,0,0))
        A%*%A12%*%A
        A
        A12%*%A%*%A12
        A12
```

```
A12%*%A
t(A12%*%A)
A%*%A12
t(A%*%A12)
```

It is a g_{12} inverse, but it is not a Moore-Penrose inverse, because neither $A^{-}A$ nor AA^{-} is symmetric.

4.13a.

```
A <- matrix(c(1,3,2,2,3,1,2,1,2,4,2,1),nrow=3)
SVDA <- svd(A)
SVDA$u%*%diag(SVDA$d)%*%t(SVDA$v)
A
```

4.13b.

```
Ainv <- t(SVDA$u%*%diag(1/SVDA$d)%*%t(SVDA$v))
A%*%Ainv%*%A
A
Ainv%*%A%*%Ainv
Ainv
Ainv%*%A
t(Ainv%*%A)
A%*%Ainv
t(A%*%Ainv)
```

It is not a Moore-Penrose inverse. It is a g_3 generalized inverse.

4.14a. B is nonnegative definite.

```
> A <- matrix(c(1,3,2,2,3,1,2,1,2,4,2,1),nrow=3)
> B <- t(A)%*%A
> B
     [,1] [,2] [,3] [,4]
[1,]   14   13    9   12
[2,]   13   14    9   15
[3,]    9    9    9   12
[4,]   12   15   12   21
> SVDB <- svd(B)
> SVDB$d
[1] 5.069044e+01 5.352440e+00 1.957116e+00 1.089186e-15
> sqrtB <- SVDB$u%*%sqrt(diag(SVDB$d))%*%t(SVDB$v)
> sqrtB%*%sqrtB
```

```
     [,1] [,2] [,3] [,4]
[1,]   14   13    9   12
[2,]   13   14    9   15
[3,]    9    9    9   12
[4,]   12   15   12   21
```

4.14b. D is positive definite.

```
> C <- matrix(c(4,2,2,2,3,1,2,1,4),nrow=3)
> D <- C%*%C
> D
     [,1] [,2] [,3]
[1,]   24   16   18
[2,]   16   14   11
[3,]   18   11   21
> SVDD <- svd(D)
> SVDD$d
[1] 50.680541  6.596539  1.722920
> sqrtD <- SVDD$u%*%sqrt(diag(SVDD$d))%*%t(SVDD$v)
> sqrtD%*%sqrtD
     [,1] [,2] [,3]
[1,]   24   16   18
[2,]   16   14   11
[3,]   18   11   21
> sqrtD
     [,1] [,2] [,3]
[1,]    4    2    2
[2,]    2    3    1
[3,]    2    1    4
```

C is the square root of D.

4.14c. F is positive definite.

```
> E <- matrix(c(4,2,2,2,-3,1,2,1,4),nrow=3)
> F <- C%*%C
> SVDF <- svd(F)
> SVDF$d
[1] 50.680541  6.596539  1.722920
> sqrtF <- SVDF$u%*%sqrt(diag(SVDF$d))%*%t(SVDF$v)
> sqrtF%*%sqrtF
     [,1] [,2] [,3]
[1,]   24   16   18
```

```
[2,]   16   14   11
[3,]   18   11   21
> sqrtF
     [,1] [,2] [,3]
[1,]    4    2    2
[2,]    2    3    1
[3,]    2    1    4
```

E is not the square root of F.

Exercises Beginning on Page 294

5.1. First, show that

$$\max_{x\neq 0}\frac{\|Ax\|}{\|x\|} = \left(\min_{x\neq 0}\frac{\|A^{-1}x\|}{\|x\|}\right)^{-1}$$

and

$$\max_{x\neq 0}\frac{\|A^{-1}x\|}{\|x\|} = \left(\min_{x\neq 0}\frac{\|Ax\|}{\|x\|}\right)^{-1}.$$

5.2a. The matrix prior to the first elimination is

$$\begin{bmatrix} 2 & 5 & 3 & 19 \\ 1 & 4 & 1 & 12 \\ 1 & 2 & 2 & 9 \end{bmatrix}.$$

The solution is $(3, 2, 1)$.

5.2b. The matrix prior to the first elimination is

$$\begin{bmatrix} 5 & 2 & 3 & 19 \\ 4 & 1 & 1 & 12 \\ 2 & 1 & 2 & 9 \end{bmatrix},$$

and x_1 and x_2 have been interchanged.

5.2c.

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 2 \end{bmatrix},$$

$$L = \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 2 & 0 \end{bmatrix},$$

$$U = \begin{bmatrix} 0 & 4 & 1 \\ 0 & 0 & 3 \\ 0 & 0 & 0 \end{bmatrix},$$

$$\rho((D+L)^{-1}U) = 1.50.$$

5.2e. $\rho((\widetilde{D}+\widetilde{L})^{-1}\widetilde{U}) = 0.9045$.

5.2g. Some R code for this is

```
tildeD <- matrix(c(2, 0, 0,
                   0, 4, 0,
                   0, 0, 2), nrow=3, byrow=T)
tildeL <- matrix(c(0, 0, 0,
                   1, 0, 0,
                   1, 2, 0), nrow=3, byrow=T)
tildeU <- matrix(c(0, 5, 3,
                   0, 0, 1,
                   0, 0, 0), nrow=3, byrow=T)
b <- c(12,19,9)

omega <- 0.1
tildeDUadj <- (1-omega)*tildeD - omega*tildeU
tildeAk <- tildeD+omega*tildeL
badj <- omega*b
xk <- c(1,1,1)
nstep <- 2
for (i in 1:nstep){
   bk <- tildeDUadj%*%xk + badj
   xkp1 <- solve(tildeAk,bk)
   dif <- sqrt(sum((xkp1-xk)^2))
   print(dif)
   xk <- xkp1
}
```

5.4a. $nm(m+1) - m(m+1)/2$. (Remember $A^{\mathrm{T}}A$ is symmetric.)

5.4g. Using the normal equations with the Cholesky decomposition requires only about half as many flops as the QR, when n is much larger than m. The QR method oftens yields better accuracy, however.

5.5a.
1. $X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}X = X$
2. $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$
3. $X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$ is symmetric (take its transpose).
4. $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}X$ is symmetric.

Therefore, $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} = X^{+}$.

5.5b. We want to show $X^{\mathrm{T}}(y - XX^{+}y) = 0$. Using the properties of X^{+}, we have

$$\begin{aligned} X^{\mathrm{T}}(y - XX^+y) &= X^{\mathrm{T}}y - X^{\mathrm{T}}XX^+y \\ &= X^{\mathrm{T}}y - X^{\mathrm{T}}(XX^+)^{\mathrm{T}}y \quad \text{because of symmetry} \\ &= X^{\mathrm{T}}y - X^{\mathrm{T}}(X^+)^{\mathrm{T}}X^{\mathrm{T}}y \\ &= X^{\mathrm{T}}y - X^{\mathrm{T}}(X^{\mathrm{T}})^+X^{\mathrm{T}}y \quad \text{property of Moore-Penrose inverses and transposes} \\ &= X^{\mathrm{T}}y - X^{\mathrm{T}}y \quad \text{property of Moore-Penrose inverses} \\ &= 0 \end{aligned}$$

Exercises Beginning on Page 314

6.1a. 1.
6.1b. 1.
6.1d. 1. (All that was left was to determine the probability that $c_n \neq 0$ and $c_{n-1} \neq 0$.)
6.2a. 11.6315.
6.3.

$$\begin{bmatrix} 3.08 & -0.66 & 0 & 0 \\ -0.66 & 4.92 & -3.27 & 0 \\ 0 & -3.27 & 7.00 & -3.74 \\ 0 & 0 & -3.74 & 7.00 \end{bmatrix}.$$

Exercises Beginning on Page 353

7.7. This is because the subspace that generates a singular matrix is a lower dimensional space than the full sample space, and so its measure is 0.
7.9. $2^{dn/2}\Gamma_d(n/2)|\Sigma|^{n/2}$.
Make the change of variables $W = 2\Sigma^{\frac{1}{2}}X\Sigma^{\frac{1}{2}}$, determine the Jacobian, and integrate.

Exercises Beginning on Page 425

8.3. 1. See Exercise 7.7, page 353.
8.10a.

$$p(c) = c^m - \alpha_1 c^{m-1} - \alpha_2\sigma_1 c^{m-2} - \alpha_3\sigma_1\sigma_2 c^{m-3} - \cdots - \alpha_m\sigma_1\sigma_2\cdots\sigma_{m-1}.$$

8.10b. Define

$$f(c) = 1 - \frac{p(c)}{c^m}.$$

This is a monotone decreasing continuous function in c, with $f(c) \to \infty$ as $c \to 0_+$ and $f(c) \to 0$ as $c \to \infty$. Therefore, there is a unique value c_* for which $f(c_*) = 1$. The uniqueness also follows from Descartes' rule of signs, which states that the maximum number of positive roots of a polynomial is the number of sign changes of the coefficients, and in the case of the polynomial $p(c)$, this is one.

8.18. This is a simple case of matrix multiplication.
To illustrate the use of R in complex matrices, I will show some code that is relevant to this problem, for a given order of course.

```
omegajn <- function(n){
#  Function to create the n^th roots of 1
    omegajn <- complex(n)
    omegajn[1] <- 1
    if (n>=2) omegajn[2] <-
          complex(re=cos(2*pi/n),im=sin(2*pi/n))
    if (n>=3) for (j in 3:n) omegajn[j] <-
                          omegajn[2]*omegajn[j-1]
    return(omegajn)
}

Fn <- function(n){
#  Function to create a Fourier matrix
    rts <- omegajn(n)
    Fn <- matrix(c(rep(1,n),rts),nrow=n)
    if (n>=3) for (j in 3:n) Fn <- cbind(Fn,rts^(j-1))
    Fn <- Fn/sqrt(n)
    return(Fn)
}

# perform multiplications to get the elementary
# circulant matrix of order 5
round(Conj(t(F5))%*%diag(rts5)%*%F5)
```

8.19. $(-1)^{\lfloor n/2 \rfloor} n^n$, where $\lfloor\cdot\rfloor$ is the floor function (the greatest integer function). For $n = 1, 2, 3, 4$, the determinants are $1, -4, -27, 256$.

Exercises Beginning on Page 506

9.3d. Assuming W is positive definite, we have

$$\widehat{\beta}_{W,C} = (X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}Wy +$$

$$(X^{\mathrm{T}}WX)^{-1}L^{\mathrm{T}}(L(X^{\mathrm{T}}WX)^{+}L^{\mathrm{T}})^{-1}(c - L(X^{\mathrm{T}}WX)^{+}X^{\mathrm{T}}Wy)\,.$$

9.8. Let $X = [X_{\mathrm{i}} \mid X_{\mathrm{o}}]$ and $Z = X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{o}} - X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}}(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1}X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}}$. Note that $X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}} = X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}}$. We have

$$
\begin{aligned}
&X_\mathrm{i}^\mathrm{T} X(X^\mathrm{T}X)^{-1}X^\mathrm{T}\\
&\quad = X_\mathrm{i}^\mathrm{T}\,[X_\mathrm{i} \mid X_\mathrm{o}] \left[\begin{array}{c|c} X_\mathrm{i}^\mathrm{T}X_\mathrm{i} & X_\mathrm{i}^\mathrm{T}X_\mathrm{o} \\ X_\mathrm{o}^\mathrm{T}X_\mathrm{i} & X_\mathrm{o}^\mathrm{T}X_\mathrm{o}\end{array}\right]^{-1} [X_\mathrm{i} \mid X_\mathrm{o}]^\mathrm{T}\\
&\quad = \left[X_\mathrm{i}^\mathrm{T}X_\mathrm{i} \mid X_\mathrm{i}^\mathrm{T}X_\mathrm{o}\right]\\
&\qquad \left[\begin{array}{c|c} (X_\mathrm{i}^\mathrm{T}X_\mathrm{i})^{-1} - (X_\mathrm{i}^\mathrm{T}X_\mathrm{i})^{-1}(X_\mathrm{o}^\mathrm{T}X_\mathrm{i})Z^{-1}(X_\mathrm{i}^\mathrm{T}X_\mathrm{o})(X_\mathrm{i}^\mathrm{T}X_\mathrm{i})^{-1} & -(X_\mathrm{i}^\mathrm{T}X_\mathrm{i})^{-1}(X_\mathrm{i}^\mathrm{T}X_\mathrm{o})Z^{-1} \\ -Z^{-1}(X_\mathrm{o}^\mathrm{T}X_\mathrm{i})(X_\mathrm{i}^\mathrm{T}X_\mathrm{i})^{-1} & Z^{-1}\end{array}\right]\\
&\qquad \begin{bmatrix} X_\mathrm{i}^\mathrm{T} \\ X_\mathrm{o}^\mathrm{T}\end{bmatrix}\\
&\quad = \left[I - (X_\mathrm{o}^\mathrm{T}X_\mathrm{i})Z^{-1}(X_\mathrm{i}^\mathrm{T}X_\mathrm{o})(X_\mathrm{i}^\mathrm{T}X_\mathrm{i})^{-1} - X_\mathrm{i}^\mathrm{T}X_\mathrm{o}Z^{-1}(X_\mathrm{o}^\mathrm{T}X_\mathrm{i})(X_\mathrm{i}^\mathrm{T}X_\mathrm{i})^{-1} \mid -X_\mathrm{i}^\mathrm{T}X_\mathrm{o}Z^{-1} + X_\mathrm{i}^\mathrm{T}X_\mathrm{o}Z^{-1}\right]\\
&\qquad \begin{bmatrix} X_\mathrm{i}^\mathrm{T} \\ X_\mathrm{o}^\mathrm{T}\end{bmatrix}\\
&\quad = X_\mathrm{i}^\mathrm{T},
\end{aligned}
$$

9.9. One possibility is

$$
\begin{bmatrix}
20 & 100\\
5 & 25\\
5 & 25\\
10 & \mathtt{NA}\\
10 & \mathtt{NA}\\
10 & \mathtt{NA}\\
\mathtt{NA} & 10\\
\mathtt{NA} & 10\\
\mathtt{NA} & 10
\end{bmatrix}.
$$

The variance-covariance matrix computed from all pairwise complete observations is

$$
\begin{bmatrix} 30 & 375 \\ 375 & 1230 \end{bmatrix},
$$

while that computed only from complete cases is

$$
\begin{bmatrix} 75 & 375 \\ 375 & 1875 \end{bmatrix}.
$$

The correlation matrix computed from all pairwise complete observations is

$$
\begin{bmatrix} 1.00 & 1.95 \\ 1.95 & 1.00 \end{bmatrix}.
$$

Note that this example is not a pseudo-correlation matrix.
In the R software system, the `cov` and `cor` functions have an argument called "`use`", which can take the values "all.obs", "complete.obs",

or "pairwise.complete.obs". The value "all.obs" yields an error if the data matrix contains any missing values. In `cov`, the values "complete.obs" and "pairwise.complete.obs" yield the variance-covariances shown above. The function `cor` with `use="pairwise.complete.obs"` yields

$$\begin{bmatrix} 1.00 & 1.00 \\ 1.00 & 1.00 \end{bmatrix}.$$

However, if `cov` is invoked with `use="pairwise.complete.obs"` and the function `cov2cor` is applied to the result, the correlations are 1.95, as in the first correlation matrix above.

9.12b. The first step is to use the trick of equation (3.97), $x^{\mathrm{T}}Ax = \mathrm{tr}(Axx^{\mathrm{T}})$, again to undo the earlier expression, and write the last term in equation (9.120) as

$$-\frac{n}{2}\mathrm{tr}\left(\Sigma^{-1}(\bar{y}-\mu)(\bar{y}-\mu)^{\mathrm{T}}\right) = -\frac{n}{2}(\bar{y}-\mu)\Sigma^{-1}(\bar{y}-\mu)^{\mathrm{T}}.$$

Now Σ^{-1} is positive definite, so $(\bar{y}-\mu)\Sigma^{-1}(\bar{y}-\mu)^{\mathrm{T}} \geq 0$ and hence is minimized for $\hat{\mu} = \bar{y}$. Decreasing this term increases the value of $l(\mu, \Sigma;\, y)$, and so $l(\hat{\mu}, \Sigma;\, y) \geq l(\mu, \Sigma;\, y)$ for all positive definite Σ^{-1}. Now, we consider the other term. Let $A = \sum_{i=1}^{n}(y_i - \bar{y})(y_i - \bar{y})^{\mathrm{T}}$. The first question is whether A is positive definite. We will refer to a text on multivariate statistics for the proof that A is positive definite with probability 1 (see Muirhead, 1982, for example). We have

$$\begin{aligned} l(\hat{\mu}, \Sigma;\, y) &= c - \frac{n}{2}\log|\Sigma| - \frac{1}{2}\mathrm{tr}\left(\Sigma^{-1}A\right) \\ &= c - \frac{n}{2}\left(\log|\Sigma| + \mathrm{tr}\left(\Sigma^{-1}A/n\right)\right). \end{aligned}$$

Because c is constant, the function is maximized at the minimum of the latter term subject to Σ being positive definite, which, as shown for expression (9.137), occurs at $\widehat{\Sigma} = A/n$.

9.14. We can develop a recursion for p_{11}^{t} based on p_{11}^{t-1} and p_{12}^{t-1},

$$p_{11}^{t} = p_{11}^{t-1}(1-\alpha) + p_{12}^{t-1}\beta,$$

and because $p_{11} + p_{12} = 1$, we have $p_{11}^{t} = p_{11}^{t-1}(1-\alpha-\beta) + \beta$. Putting this together, we have

$$\lim_{t\to\infty} P = \begin{bmatrix} \beta/(\alpha+\beta) & \alpha/(\alpha+\beta) \\ \beta/(\alpha+\beta) & \alpha/(\alpha+\beta) \end{bmatrix},$$

and so the limiting (and invariant) distribution is $\pi_s = (\beta/(\alpha + \beta),\ \alpha/(\alpha+\beta))$.

9.15c. From the exponential growth, we have $N^{(T)} = N^{(0)}\mathrm{e}^{rT}$; hence,

$$r = \frac{1}{T}\log\left(N^{(T)}/N^{(0)}\right) = \frac{1}{T}\log(r_0).$$

9.23. This is an open question. If you get a proof of convergence, submit it for publication. You may wish to try several examples and observe the performance of the intermediate steps. I know of no case in which the method has not converged.

9.24b. Starting with the correlation matrix given above as a possible solution for Exercise 9.9, four iterations of equation (9.81) using $\delta = 0.05$ and $f(x) = \tanh(x)$ yield

$$\begin{bmatrix} 1.000 & 0.997 \\ 0.997 & 1.000 \end{bmatrix}.$$

Exercises Beginning on Page 573

10.1a. The computations do not overflow. The first floating-point number x such that $x + 1 = x$ is

$$0.10 \cdots 0 \times b^{p+1}.$$

Therefore, the series converges at the value of i such that $i(i+1)/2 = x$. Now solve for i.

10.2. The function is $\log(n)$, and Euler's constant is 0.57721....

10.5. 2^{-56}. (The standard has 53 bits normalized, so the last bit is 2^{-55}, and half of that is 2^{-56}.)

10.6a. Normalized: $2b^{p-1}(b-1)(e_{\max} - e_{\min} + 1) + 1$.
Nonnormalized: $2b^{p-1}(b-1)(e_{\max} - e_{\min} + 1) + 1 + 2b^{p-1}$.

10.6b. Normalized: $b^{e_{\min}-1}$.
Nonnormalized: $b^{e_{\min}-p}$.

10.6c. $1 + b^{-p+1}$ or $1 + b^{-p}$ when $b = 2$ and the first bit is hidden.

10.6d. b^p.

10.6e. 22.

10.11. First of all, we recognize that the full sum in each case is 1. We therefore accumulate the sum from the direction in which there are fewer terms. After computing the first term from the appropriate direction, take a logarithm to determine a scaling factor, say s^k. (This term will be the smallest in the sum.) Next, proceed to accumulate terms until the sum is of a different order of magnitude than the next term. At that point, perform a scale adjustment by dividing by s. Resume summing, making similar scale adjustments as necessary, until the limit of the summation is reached.

10.13. The result is close to 1.
What is relevant here is that numbers close to 1 have only a very few digits of accuracy; therefore, it would be better to design this program so that it returns $1 - \Pr(X \leq x)$ (the "significance level"). The purpose and the anticipated use of a program determine how it should be designed.

10.16a. 2.

10.16b. 0.

10.16c. No (because the operations in the "for" loop are not chained).

10.17c.

```
$a = x_1$
$b = y_1$
$s = 0$
for $i = 2, n$
{
  $d = (x_i - a)/i$
  $e = (y_i - b)/i$
  $a = d + a$
  $b = e + b$
  $s = i(i-1)de + s$
}.
```

10.20. 1. No; 2. Yes; 3. No; 4. No.

10.21. A very simple example is

$$\begin{bmatrix} 1 & 1+\epsilon \\ 1 & 1 \end{bmatrix},$$

where $\epsilon < b^{-p}$, because in this case the matrix stored in the computer would be singular. Another example is

$$\begin{bmatrix} 1 & a(1+\epsilon) \\ a(1+\epsilon) & a^2(1+2\epsilon) \end{bmatrix},$$

where ϵ is the machine epsilon.

10.23a.

```
> 1/0
[1] Inf
> -5*(1/0)
[1] -Inf
> (1/0)*(1/0)
[1] Inf
> 1/0==5*(1/0)
[1] TRUE
```

10.23b.

```
> 0/0
[1] NaN
> -5*(0/0)
[1] NaN
> 0/0==NaN
[1] NA
```

```
> x <- NaN
> x==NaN
[1] NA
> is.nan(x)
[1] TRUE
> is.na(x)
[1] TRUE
```

10.23c.

```
> x<-NA
> x==NA
[1] NA
> is.nan(x)
[1] FALSE
> is.na(x)
[1] TRUE
```

10.26. By decomposing 24 and multiplying some factors, or by using Stirling's approximation, we know that there are approximately 24 or 25 digits in 24! By considering the number of times 10 occurs as a factor, we know that the last four digits are 0s.

```
> x24mpfr <- mpfr(24, prec=25*log2(10))
> x24mpfrfac <- factorial(x24mpfr)
> print(x24mpfrfac, digits=26)
1 'mpfr' number of precision  83   bits
[1] 620448401733239439360000
```

For comparison,

```
> x24 <- 24
> x24fac <- factorial(x24)
> print(x24fac, digits=22)
[1] 6.204484017332394e+23
```

Exercises Beginning on Page 592

11.2a. $\mathrm{O}(nk)$.

11.2c. At each successive stage in the fan-in, the number of processors doing the additions goes down by approximately one-half.

If $p \approx k$, then $\mathrm{O}(n \log k)$ (fan-in on one element of c at a time)
If $p \approx nk$, then $\mathrm{O}(\log k)$ (fan-in on all elements of c simultaneously)
If p is a fixed constant smaller than k, the order of time does not change; only the multiplicative constant changes.
Notice the difference in the order of time and the order of the number of computations. Often there is very little that can be done about the order of computations.

11.2d. Because in a serial algorithm the magnitudes of the summands become more and more different. In the fan-in, they are more likely to remain relatively equal. Adding magnitudes of different quantities results in benign roundoff, but many benign roundoffs become bad. (This is not catastrophic cancellation.) Clearly, if all elements are nonnegative, this argument would hold. Even if the elements are randomly distributed, there is likely to be a drift in the sum (this can be thought of as a random walk). There is *no difference* in the number of computations.

11.2e. Case 1: $p \approx n$. Give each c_i a processor – do an outer loop on each. This would likely be more efficient because all processors are active at once.
Case 2: $p \approx nk$. Give each $a_{ij}b_j$ a processor – fan-in for each. This would be the same as the other.
If p is a fixed constant smaller than n, set it up as in Case 1, using n/p groups of c_i's.

11.2f. If $p \approx n$, then $\mathrm{O}(k)$.
If $p \approx nk$, then $\mathrm{O}(\log k)$.
If p is some small fixed constant, the order of time does not change; only the multiplicative constant changes.

Exercises Beginning on Page 633

12.2. Here is a recursive Matlab function for the Strassen algorithm due to Coleman and Van Loan. When it uses the Strassen algorithm, it requires the matrices to have even dimension.

```
   function C = strass(A,B,nmin)
%
% Strassen matrix multiplication C=AB
%       A, B must be square and of even dimension
% From Coleman and Van Loan
% If n <= nmin, the multiplication is done conventionally
%
   [n n ] = size(A);
   if n <= nmin
      C = A * B;   % n is small, get C conventionally
   else
      m = n/2; u = 1:m; v = m+1:n;
      P1 = strass(A(u,u)+A(v,v), B(u,u)+B(v,v), nmin);
```

```
    P2 = strass(A(v,u)+A(v,v), B(u,u),        nmin);
    P3 = strass(A(u,u),        B(u,v)-B(v,v), nmin);
    P4 = strass(A(v,v),        B(v,u)-B(u,u), nmin);
    P5 = strass(A(u,u)+A(u,v), B(v,v),        nmin);
    P6 = strass(A(v,u)-A(u,u), B(u,u)+B(u,v), nmin);
    P7 = strass(A(u,v)-A(v,v), B(v,u)+B(v,v), nmin);
    C = [P1+P4-P5+P7 P3+P5; P2+P4 P1+P3-P2+P6];
 end
```

12.5a.

```
    real a(4,3)
    data a/3.,6.,8.,2.,5.,1.,6.,3.,6.,2.,7.,1./
    n = 4
    m = 3
    x1 = a(2,2) ! Temporary variables must be used because of
    x2 = a(4,2) ! the side effects of srotg.
    call srotg(x1, x2,, c, s)
    call srot(m, a(2,1), n, a(4,1), n, c, s)
    print *, c, s
    print *, a
    end
```

This yields 0.3162278 and 0.9486833 for c and s. The transformed matrix is

$$\begin{bmatrix} 3.000000 & 5.000000 & 6.000000 \\ 3.794733 & 3.162278 & 1.581139 \\ 8.000000 & 6.000000 & 7.000000 \\ -5.059644 & -0.00000002980232 & -1.581139 \end{bmatrix}.$$

12.7b. Using the Matrix package in R, after initializing `rho` and `sig2`, this is

```
Vinv <- sparseMatrix(i=c(1,1,2,2,2,3,3,3,4,4,4,5,5,5,6,6,6,7,7,7,8,8,8,9,
                                                                9,9,10,10),
                     j=c(1,2,1:3,2:4,3:5,4:6,5:7,6:8,7:9,8:10,9,10),
                     x=c(1,-rho,rep(c(-rho,1+rho^2,-rho),8),-rho,1))/
                                 (1-rho^2)*sig2
```

12.9. 10.7461941829033 and 10.7461941829034.

12.11.

$$\begin{matrix} \frac{-(fh)+ei}{-(ceg)+bfg+cdh-afh-bdi+aei} & \frac{ch-bi}{-(ceg)+bfg+cdh-afh-bdi+aei} & \frac{-(ce)+bf}{-(ceg)+bfg+cdh-afh-bdi+aei} \\ \frac{fg-di}{-(ceg)+bfg+cdh-afh-bdi+aei} & \frac{-(cg)+ai}{-(ceg)+bfg+cdh-afh-bdi+aei} & \frac{cd-af}{-(ceg)+bfg+cdh-afh-bdi+aei} \\ \frac{-(eg)+dh}{-(ceg)+bfg+cdh-afh-bdi+aei} & \frac{bg-ah}{-(ceg)+bfg+cdh-afh-bdi+aei} & \frac{-(bd)+ae}{-(ceg)+bfg+cdh-afh-bdi+aei} \end{matrix}$$