

GRADES:

```

8| 0 2 3
7| 5
6| 1 5 5 6 7 8 9
5| 3 6 7
4|
3| 1

```

1. (12 points) In a regression problem where  $X$  is  $(n \times m)$ , show that the vector  $\hat{y}$  of 'fitted values' can be written as  $\hat{y} = Hy$ , for some  $(n \times n)$  matrix  $H$  and find  $H$ . Using the fact that  $\text{tr}(AB) = \text{tr}(BA)$ , show that the average value of the diagonal elements of  $H$  is  $m/n$  (the trace of a square matrix is the sum of its diagonal elements). Express  $H_{ii}$  as a function of the elements of the matrices you get from the Modified Gram Schmidt Decomposition of  $X$ .

**SOLUTION:**

By definition  $\hat{y} = X\hat{\beta}$  and since  $\hat{\beta} = (X^T X)^{-1} X^T y$ , we have  $\hat{y} = X(X^T X)^{-1} X^T y$ , and so  $H = X(X^T X)^{-1} X^T$ . If we let  $A = X(X^T X)^{-1}$  and  $B = X^T$ , then  $\text{tr}(AB) = \text{tr}(BA)$  gives the sum of the  $n$  diagonal elements of  $H$  as

$$\text{tr}(H) = \text{tr}(X^T X (X^T X)^{-1}) = \text{tr}(I_m) = m,$$

and so the 'average hat diagonal' is  $m/n$ . Finally, the MGS gives  $X = QR$  where  $Q^T Q = D$ , so

$$H = QR(R^T Q^T QR)^{-1} R^T Q^T = QR(R^T DR)^{-1} R^T Q^T = QRR^{-1} D^{-1} R^{-T} R^T Q^T = QD^{-1} Q^T,$$

which gives

$$H_{ii} = \sum_{j=1}^m Q_{ij}^2 / D_{jj}.$$

2. (8 points) Show that a kernel density estimator is a pdf.

**SOLUTION:**

We defined a kernel density estimator for data  $x_1, \dots, x_n$  as

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right),$$

where  $K$  is a weight function called a kernel which we assumed to be a pdf. Thus making the change of variables  $z = (x - x_i)/h$  (which gives  $dx = h dz$ ) we have

$$\begin{aligned} \int_{-\infty}^{\infty} \hat{f}(x) dx &= \frac{1}{nh} \int_{-\infty}^{\infty} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right) dx = \frac{1}{nh} \sum_{i=1}^n \int_{-\infty}^{\infty} K\left(\frac{x - x_i}{h}\right) dx \\ &= \frac{1}{nh} \sum_{i=1}^n \int_{-\infty}^{\infty} h K(z) dz = \frac{1}{nh} \sum_{i=1}^n h = \frac{nh}{nh} = 1. \end{aligned}$$

---

3. (10 points) Find a simple recursion for the log of Poisson probabilities  $f(x)$  for  $x \geq 0$  and use this to write a fortran subroutine that will return  $f(x)$  for  $x$  up to  $M$  for user specified  $M$  and Poisson mean value parameter  $\lambda$ .

**SOLUTION:**

For  $x \geq 0$ , Poisson probabilities are given by  $f(x) = \lambda^x e^{-\lambda} / x!$ , which gives for  $x \geq 0$ ,

$$\frac{f(x+1)}{f(x)} = \frac{\lambda^{x+1} e^{-\lambda}}{(x+1)!} \frac{x!}{\lambda^x e^{-\lambda}} = \frac{\lambda}{x+1},$$

and thus  $\log f(x+1) = \log f(x) + \log \lambda - \log(x+1)$ , and so we have the following subroutine

```

      subroutine dpoiss(lambda,m,f)
c*****
c
c   Subroutine to find poisson probabilities for x = 0, ..., m
c   for parameter lambda. The probabilities are returned in the
c   vector f and lambda and f are double precision.
c
c*****

      double precision lambda,f(1),c,all

      c=-lambda
      all=log(lambda)
      f(1)=exp(c)

      do 10 i=1,m
      c=c+all-log(dble(i))
10    f(i+1)=exp(c)

      return
      end

```

---

4. (12 points) For a nonlinear regression problem having mean function  $f$ , express the gradient and Hessian in terms of the Jacobian, residual vector, and the matrices of second derivatives of  $f$  with respect to parameters. What is the difference between Newton-Raphson and Gauss-Newton in this situation?

**SOLUTION:**

In this case, it is easy to derive the fact that the elements of the gradient and Hessian can be written as

$$g_j = \frac{\partial S}{\partial \theta_j} = -2 \sum_{i=1}^n (y_i - f_i) \frac{\partial f_i}{\partial \theta_j}$$

$$H_{jk} = \frac{\partial^2 S}{\partial \theta_j \partial \theta_k} = -2 \sum_{i=1}^n \left[ (y_i - f_i) \frac{\partial^2 f_i}{\partial \theta_j \partial \theta_k} - \frac{\partial f_i}{\partial \theta_j} \frac{\partial f_i}{\partial \theta_k} \right],$$

where  $f_i = f(X_i; \theta)$ . If we define the  $(n \times r)$  Jacobian matrix  $J$  to have  $(i, j)$ th element  $\partial f_i / \partial \theta_j$ , and the  $(r \times r)$  matrices  $G_1, \dots, G_n$  so that the  $(j, k)$ th element of  $G_i$  is  $\partial^2 f_i^2 / \partial \theta_j \partial \theta_k$ , and finally

the vector  $r$  of length  $n$  to have  $i$ th element  $r_i = y_i - f_i$ , then the gradient and Hessian are given by

$$g = -2J^T r, \quad H = 2J^T J - 2 \sum_{i=1}^n r_i G_i.$$

Often, the Hessian is approximated by the first term in this sum, which gives what is called the Gauss-Newton algorithm.

**5.** (12 points) If from a regression problem all I wanted were least squares estimators for coefficients, which method has more numerical operations, the MCD or the MGS?

**SOLUTION:**

To do the MCD method, we must first find  $X^T X$  (which takes  $m(m+1)n/2$  operations), then do the MCD, that is, find  $L$  and  $D$ , and then solve two triangular systems. To do the MGS, we don't need to get  $X^T X$ . We must find  $Q$  and  $R$  (which I told you in class is  $m^2n/2$ ), and then solve basically the same two triangular systems as in the MCD method. Thus the only question is how many operations getting  $L$  and  $D$  take. I expected you to recall that getting the  $(i, j)$ th element of  $L$  takes basically  $j$  operations. If you add up  $\sum_{i=1}^m \sum_{j=1}^i j$  you get basically  $m^3/6$ . Thus ignoring triangular system solving, MCD takes roughly  $m^2n/2 + m^3/6$  while MGS takes  $m^2n/2$ , and MGS is slightly faster.

**6.** (8 points) What is the relative absolute error in using the composite Simpson's rule (with four intervals) to approximate  $\int_0^1 \cos \pi x dx$ ?

**SOLUTION:**

The exact value of the integral can be obtained as

$$\int_0^1 \cos \pi x dx = -\frac{\sin \pi x}{\pi} \Big|_{x=0}^{x=1} = 0,$$

while the composite Simpson's Rule gives (with  $h = 1/4$ ),

$$\begin{aligned} \int_0^1 \cos \pi x dx &\doteq \frac{h}{3} [f(0) + 4f(1/4) + 2f(1/2) + 4f(3/4) + f(1)] \\ &= \frac{1}{12} [1 + 4f(1/4) + 0 - 4f(1/4) - 1] \\ &= 0, \end{aligned}$$

and so there is no error.

**7.** (8 points) In constructing a histogram of data  $X_1, \dots, X_n$ , how can I find the frequencies without using any if statements?

**SOLUTION:**

Given lower and upper limits  $a$  and  $b$  such that none of the  $X$ 's are less than  $a$  or greater than  $b$ , we can get frequencies for  $M$  equally wide intervals from  $a$  to  $b$  by initializing a vector `ind` of length  $M$  to be zeros, and then for  $i$  from 1 to  $n$  we calculate

$$j = \left[ \frac{X_i - a}{b - a} \right] + 1,$$

where  $[x]$  denotes the greatest integer less than or equal to  $x$ , and increment the  $j$ th element of `ind`.

8. (10 points) Why does the program below print

```
n =          -2

      n=-123456
      call prtsc(n)
      stop
      end

      subroutine prtsc(n)
      integer*2 n
      write(*,10) n
10    format(' n = ',i10)
      return
      end
```

**SOLUTION:**

The problem is that in the main program `n` is by default a 4 byte integer whose representation is obtained by 2's complement arithmetic as

0000 0000 0000 0001 1110 0010 0100 0000	(binary representation of 123456)
1111 1111 1111 1110 0001 1101 1011 1111	(2's complement)
<div style="text-align: right; padding-right: 10px;">1</div> -----	(add one)
1111 1111 1111 1110 0001 1101 1100 0000	(result)

since  $123456 = 2^{16} + 2^{15} + 2^{14} + 2^{13} + 2^9 + 2^6$ .

In the subroutine, `n` is a 2 byte integer which will receive 2 of the 4 bytes from the main program. Note that the 2 bytes on the left above is in fact the 2 byte representation of `-123456` and thus now we know which 2 bytes the subroutine gets!

9. (10 points) To generate  $n$  pairs of iid  $N(0,1)$  random variables using the usual normal random number generator, how many pairs of  $U(0,1)$ 's do you need to generate to be 95% sure that you have enough to generate the normals?

**SOLUTION:**

To get a pair of normals, we must first generate pairs  $(U_1, U_2)$  of  $U(-1, 1)$ 's until  $U_1^2 + U_2^2 < 1$ , that is until we get a point uniformly distributed in the unit square to also be in the unit circle.

The probability of this is the ratio of the areas of the circle and the square, that is,  $p = \pi/4$ . Thus the total number of pairs of uniforms needed is the random variable

$$X = \sum_{i=1}^n Z_i,$$

where  $Z_i$  is the number of uniforms needed to get the  $i$ th pair of uniforms. Note that  $X$  has the negative binomial distribution with parameters  $n$  and  $p$  and one could write a program to find the 95th percentile of this distribution. If  $n$  is large, since  $X$  is the sum of the i.i.d. random variables  $Z_i$ , then by the central limit theorem,  $X$  is approximately normal with mean  $n$  times the mean of  $Z_i$  (which is  $1/p$ ) and variance  $n$  times the variance of  $Z_i$  (which is  $(1-p)/p^2$ ), from which we can find an approximation to the 95th percentile of  $X$  as  $np + 1.645\sqrt{n(1-p)/p^2}$ .

---

**10.** (10 points) In quick sort, what would be the result of the first splitting of the array

4 13 11 6 2 10 5 1 3 8 7 12 9?

**SOLUTION:**

We start by sorting the first, last, and middle values (the 4, 5, and 9) which are already in order. Then we temporarily swap the 4 and the 5. We can then swap the 13 and the 3 (the first value larger than 5 starting from the left and the first value smaller than the 5 from the right). We can then similarly swap the 11 and the 1, and then the 6 and the 4, at which point there are no more pairs to swap. The pointer is at the 10 which is bigger than 5 so we back up one to the 2 and finish by swapping the 5 and the 2, giving:

2 3 1 4 5 10 6 11 13 8 7 12 9.

---