Detailed Solutions A-07 JUNE 2003

Q1. a. C $I = \pi f(0)$, where $f(x) = \cos x$. $I = \pi = 3.14$.

b. B

c. C It is a property of Gauss formulas.

d. A

e. A Use divided difference interpolation.

f. A If $\boldsymbol{Ax} = \lambda \boldsymbol{x}$, then $\boldsymbol{A}^{-1}\boldsymbol{x} = (1/\lambda)\boldsymbol{x}$.

g k = 4 + (4 > 1) = 4 + 1 = 5.

h D $\varepsilon_{n+1} = C\,\varepsilon_n^2$. Order $= 2$.

PART I

Q2 $\lim_{x\to\infty} x_{n+1} = x = \lim_{x\to\infty} x_n$. The error is defined by $x_n - \xi = \varepsilon_n$.

(i) $x = \frac{x}{2}(1+\frac{a}{x^2})$ gives $2x^2 = x^2 + a$, or $x = \sqrt{a}$.

(ii) $x = \frac{x}{2}(3-\frac{x^2}{a})$ gives $2 = 3 - \frac{x^2}{a}$, or $x = \sqrt{a}$.

Substituting $x_n = \xi + \varepsilon_n$ in the first formula, we get

$$\xi + \varepsilon_{n+1} = \frac{1}{2}(\xi+\varepsilon_n)\left[1+\left(1+\frac{\varepsilon_n}{\xi}\right)^{-2}\right]$$

$$= \frac{1}{2}(\xi+\varepsilon_n)\left[2-2\left(\frac{\varepsilon_n}{\xi}\right)+3\left(\frac{\varepsilon_n}{\xi}\right)^2-4\left(\frac{\varepsilon_n}{\xi}\right)^3+\cdots\right]$$

$$= \frac{1}{2}\left[2\xi+\frac{\varepsilon_n^2}{\xi}-\frac{\varepsilon_n^3}{\xi^2}+\cdots\right]$$

Hence, $\varepsilon_{n+1} = \frac{\varepsilon_n^2}{2\xi} - \frac{\varepsilon_n^3}{2\xi^2} + \cdots$ (1)

or $\varepsilon_{n+1} \approx \frac{\varepsilon_n^2}{2\xi}$. (2)

Substituting $x_n = \xi + \varepsilon_n$ in the second formula, we get

$$\xi + \varepsilon_{n+1} = \frac{1}{2}(\xi+\varepsilon_n)\left[3-\frac{1}{\xi^2}(\xi+\varepsilon_n)^2\right] = \frac{1}{2}\left[2\xi - \frac{3\varepsilon_n^2}{\xi} - \frac{\varepsilon_n^3}{\xi^2}\right]$$

Hence, $\varepsilon_{n+1} = -\frac{3\varepsilon_n^2}{2\xi} - \frac{\varepsilon_n^3}{2\xi^2}$ (3)

or $\varepsilon_{n+1} \approx -\frac{3\varepsilon_n^2}{2\xi}$ (4)

Error in the second formula given by (4) is 3 times the error in the first formula given by (2). If, multiply the first formula by 3 and add to the second formula, then from (1) and (3)

$$\varepsilon_{n+1} = -\frac{2\varepsilon_n^3}{\xi^2} + \cdots$$

Then, the order of the new formula is 3. The new formula is

$$4x_{n+1} = \frac{1}{2}x_n\left[3 + \frac{3a}{x_n^2} + 3 - \frac{x_n^2}{a}\right] \quad or \quad x_{n+1} = \frac{1}{8}x_n\left[6 + \frac{3a}{x_n^2} - \frac{x_n^2}{a}\right] .$$

Q 3a

$$A = \begin{bmatrix} 3 & 2 & 2 \\ 2 & 4 & 4 \\ 2 & 4 & 5 \end{bmatrix} = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}\begin{bmatrix} u_{11} & 0 & 0 \\ u_{12} & u_{22} & 0 \\ u_{13} & u_{23} & u_{33} \end{bmatrix}$$

Comparing the elements, we get

$u_{33}^2 = 5,\ u_{33} = \sqrt{5};\quad u_{13}u_{33} = 2,\ u_{13} = 2/\sqrt{5};\quad u_{23}u_{33} = 4,\ u_{23} = 4/\sqrt{5};$

$u_{22}^2 + u_{23}^2 = 4,\ u_{22} = 2/\sqrt{5};\quad u_{12}u_{22} + u_{13}u_{23} = 2,\ u_{12} = 1/\sqrt{5};\quad u_{11}^2 + u_{12}^2 + u_{13}^2 = 3,\ u_{11} = \sqrt{2}.$

Hence, $U = \begin{bmatrix} \sqrt{2} & 1/\sqrt{5} & 2/\sqrt{5} \\ 0 & 2/\sqrt{5} & 4/\sqrt{5} \\ 0 & 0 & \sqrt{5} \end{bmatrix}$

Q 3b

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
void main()
{
        clrscr();
        float   a[10][10], b[10], x[10], oldx[10], sum, big, c;
        float   eps;
        int     n, niter, i, j, ii, jj, k, l;
        printf("Input the order of matrix : n\n");
        printf("Input the number of iterations : niter\n");
        printf("Input error tolerance : eps\n");
        scanf("%d %d %e", &n, &niter, &eps);
        printf("n = %d, niter = %d, eps = %e\n", n, niter, eps);
        printf("Input augmented matrix row-wise\n");
        printf("Elements of the augmented matrix\n");
        for (i = 1;  i <= n; i++)
        {
                for (j = 1; j <= n; j++)
                {
                        scanf("%f", &a[i][j]);
                        printf("%f  ", a[i][j]);
                }
                scanf("%f", &b[i]);
                printf("      %f\n", b[i]);
        }
        printf("Input initial approx. to the solution vector\n");
        for (i = 1;  i <= n; i++)
        {
                scanf("%f", &oldx[i]);
                printf("%f   ", oldx[i]);
        }
        printf("\n");
        for (i = 1;  i <= n; i++)
```

```
        x[i] = oldx[i];
        for (ii = 1;  ii <= niter; ii++)
        {
                for (i = 1; i <= n; i++)
                {
                        sum = 0.0;
                        for (j = 1;  j <= n; j++)
                        {
                                if((i - j) != 0)
                                        sum = sum + a[i][j] * oldx[j];
                        }
                        x[i] = (b[i] - sum) / a[i][i];
                }
        big = fabs(x[1] - oldx[1]);
        for (k = 2; k <= n; k++)
        {
                c = fabs(x[k] - oldx[k]);
                if(c > big)
                big = c;
        }
        if(big <= eps)
                goto l10;
        for (l = 1;  l <= n; l++)
                oldx[l] = x[l];
        }
        printf("ITERATIONS NOT SUFFICIENT\n");
        printf("Solution vector at this stage\n");
        for (i = 1; i <= n; i++)
                printf("  %f ", x[i]);
        printf("\n");
        goto l20;
l10:    printf("Number of iterations = %d\n", ii);
        printf("Solution vector\n");
        for (i = 1; i <= n; i++)
                printf("  %f ", x[i]);
        printf("\n");
l20:
}
```

Q 4a Let the uniform mesh be defined as $x_0, x_1, \ldots, x_n, x_{n+1}, \ldots$ where $x_i = x_0 + ih, i = 1, 2, \ldots$ Integrate the given differential equation $y' = f(x, y)$ in interval $[x_n, x_{n+1}]$ to get

$$\int_{x_n}^{x_{n+1}} \frac{dy}{dx}\,dx = \int_{x_n}^{x_{n+1}} f(x, y)\,dx \quad or \quad y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(x, y)\,dx$$

We note that $f(x, y)$ is the slope of the solution curve and it changes continuously. Approximate the changing slope in $[x_n, x_{n+1}]$ by the fixed slope $f(x_n, y_n)$ at $x = x_n$. Then, we get the approximation

$$y_{n+1} = y_n + f(x_n, y_n)\int_{x_n}^{x_{n+1}} dx \quad or \quad y_{n+1} = y_n + h\,f(x_n, y_n)$$

where $x_{n+1} - x_n = h$. This method is called the Euler method.

We have, $x_0 = 0$, $y_0 = 1$ and $y_{n+1} = y_n + 0.1(x_n + y_n)$, $n = 0, 1, 2, 3$.

For $n = 0$, we get $y(0.1) \approx y_1 = y_0 + 0.1(0 + 1) = 1.1$.

For $n = 1$, we get $y(0.2) \approx y_2 = y_1 + 0.1(x_1 + y_1) = 1.1 + 0.1(0.1 + 1.1) = 1.22$.

For $n = 2$, we get $y(0.3) \approx y_3 = y_2 + 0.1(x_2 + y_2) = 1.22 + 0.1(0.2 + 1.22) = 1.362$.

For $n = 3$, we get $y(0.4) \approx y_4 = y_3 + 0.1(x_3 + y_3) = 1.362 + 0.1(0.3 + 1.362) = 1.5282$.

Q 4b Gauss-Jordan method gives $[A \,|\, I\,] \to [I \,|\, A^{-1}]$.

$$\left[\begin{array}{ccc|ccc} 1 & 1 & 2 & 1 & 0 & 0 \\ 1 & 5 & 8 & 0 & 1 & 0 \\ 2 & 8 & 14 & 0 & 0 & 1 \end{array}\right], \quad R_2 - R_1,\ R_3 - 2R_1, \quad \left[\begin{array}{ccc|ccc} 1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 4 & 6 & -1 & 1 & 0 \\ 0 & 6 & 10 & -2 & 0 & 1 \end{array}\right], \quad R_2/4,$$

$$\left[\begin{array}{ccc|ccc} 1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 3/2 & -1/4 & 1/4 & 0 \\ 0 & 6 & 10 & -2 & 0 & 1 \end{array}\right], \quad R_1 - R_2,\ R_3 - 6R_2, \quad \left[\begin{array}{ccc|ccc} 1 & 0 & 1/2 & 5/4 & -1/4 & 0 \\ 0 & 1 & 3/2 & -1/4 & 1/4 & 0 \\ 0 & 0 & 1 & -1/2 & -3/2 & 1 \end{array}\right],$$

$$R_1 - \frac{R_3}{2},\ R_2 - \frac{3R_3}{2}, \quad \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 3/2 & 1/2 & -1/2 \\ 0 & 1 & 0 & 1/2 & 5/2 & -3/2 \\ 0 & 0 & 1 & -1/2 & -3/2 & 1 \end{array}\right]. \quad A^{-1} = \left[\begin{array}{ccc} 3/2 & 1/2 & -1/2 \\ 1/2 & 5/2 & -3/2 \\ -1/2 & -3/2 & 1 \end{array}\right].$$

Q 4c Let the linear polynomial be $P_1(x) = c_0 + c_1\, x$. We have $\sum x_i = -2, \quad \sum x_i^2 = 6,.$ $\sum f(x_i) = 13, \quad \sum x_i f(x_i) = -13$. Normal equations for fitting the required linear polynomial are (with w(x)=1):

$$\sum_{i=0}^{3} f(x_i) - \sum_{i=0}^{3} c_0 - \sum_{i=0}^{3} c_1\, x_i = 0$$

$$\sum_{i=0}^{3} f(x_i)\, x_i - \sum_{i=0}^{3} c_0\, x_i - \sum_{i=0}^{3} c_1\, x_i^2 = 0$$

This gives $13 - 4\, c_0 + 2\, c_1 = 0$ and $-13 + 2\, c_0 - 6\, c_1 = 0$. Hence we get $c_0 = -13/10, \ c_1 = 13/5$. Hence the required linear polynomial is $P_1(x) = \frac{13}{10}(2\,x - 1)$.

Q 5 Power method : Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be the distinct eigen values such $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$ and $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ be the corresponding eigenvectors. Then, any vector $\mathbf{v}$ in this vector space of eigenvectors can be written as

$$\mathbf{v} = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \ldots + c_n\mathbf{v}_n .$$

Then, $\mathbf{A}\mathbf{v} = \mathbf{A}\,(c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \ldots + c_n\mathbf{v}_n) = c_1\lambda_1\mathbf{v}_1 + c_2\lambda_2\mathbf{v}_2 + \ldots + c_n\lambda_n\mathbf{v}_n$

$$= \lambda_1\left[c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)\mathbf{v}_2 + \ldots + c_n\left(\frac{\lambda_n}{\lambda_1}\right)\mathbf{v}_n\right]$$

Pre-multiplying by **A**, k–1 times and k times, we get

$$\mathbf{A}^k\mathbf{v} = \lambda_1^k\left[c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^k\mathbf{v}_2 + \ldots + c_n\left(\frac{\lambda_n}{\lambda_1}\right)^k\mathbf{v}_n\right] \qquad (1)$$

$$\mathbf{A}^{k+1}\mathbf{v} = \lambda_1^{k+1}\left[c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^{k+1}\mathbf{v}_2 + \ldots + c_n\left(\frac{\lambda_n}{\lambda_1}\right)^{k+1}\mathbf{v}_n\right] \qquad (2)$$

Now, $|\lambda_i/\lambda_1| < 1$, $i = 2, \ldots, n$. Hence, as $k \to \infty$, the right hand sides of (1) and (2) tend to $\lambda_1^k c_1\mathbf{v}_1$ and $\lambda_1^{k+1} c_1\mathbf{v}_1$. Therefore, all the ratios of the components of the left hand sides of (1) and (2) tend to λ_1, the largest eigen value in magnitude

$$\lambda_1 = \lim_{k\to\infty}\left[\frac{\left(A^{k+1}\mathbf{v}\right)_r}{\left(A^{k}\mathbf{v}\right)_r}\right], \quad r = 1, 2, \cdots, n.$$

The iteration is stopped when the magnitudes of the differences of the ratios are ≤ given error tolerance. The algorithm is given by

$$\mathbf{y}_{k+1} = \mathbf{A}\mathbf{v}_k, \qquad \mathbf{v}_{k+1} = \mathbf{y}_{k+1}/m_{k+1}, \qquad m_{k+1} = \max_r\left|\left(\mathbf{y}_{k+1}\right)_r\right|$$

$$\lambda_1 = \lim_{k\to\infty}\left[\frac{\left(\mathbf{y}_{k+1}\right)_r}{\left(\mathbf{v}_k\right)_r}\right], \quad r = 1, 2, \ldots, n.$$

$\mathbf{v}_{k+1}$ is the corresponding eigen vector. We have the following results.

$\mathbf{v}_0 = [1, \quad 1, \quad 1]^T, \quad \mathbf{y}_1 = \mathbf{A}\mathbf{v}_0 = [28, \quad 4, \quad -2]^T, \quad \mathbf{v}_1 = [1, \quad 0.14286, \quad -0.07143]^T,$

$\mathbf{y}_2 = \mathbf{A}\mathbf{v}_1 = [25, \ 1.42858, \ 2.28572]^T, \quad \mathbf{v}_2 = [1, \quad 0.05714, \quad 0.09143]^T,$

$\mathbf{y}_3 = \mathbf{A}\mathbf{v}_2 = [25.24, \ 1.17142, \ 1.63428]^T, \ \mathbf{v}_3 = [1, \quad 0.04641, \quad 0.06475]^T,$

$\mathbf{y}_4 = \mathbf{A}\mathbf{v}_3 = [25.17591, \ 1.13923, \ 1.741]^T, \ \mathbf{v}_4 = [1, \quad 0.04525, \quad 0.06915]^T,$

$\mathbf{y}_5 = \mathbf{A}\mathbf{v}_4 = [25.5908, \ 1.13575, \ 1.7234]^T, \ \mathbf{v}_5 = [1, \quad 0.04438, \quad 0.06734]^T.$

The ratios of the components of $\mathbf{y}_5$ and $\mathbf{v}_4$ are $[25.5908, 25.0994, 24.9226]^T$. The three ratios tend to the same eigen value 25 which gives $|\lambda_1|$. The corresponding eigen vector is $\mathbf{v}_5$.

Q 6a The bound for the error in quadratic interpolation is given by

$$|Error| \le \frac{1}{6}\max\left|(x-x_0)(x-x_1)(x-x_2)\right| M_3$$

where $M_3 = \max_{x_0\le x\le x_2}|f'''(x)|$.

Since the data is equispaced, assume without any loss of generality that $x_0 = -h, x_1 = 0, x_2 = h$. Then,

$$\max\left|(x-x_0)(x-x_1)(x-x_2)\right| \le \max\left|x\left(x^2-h^2\right)\right|.$$

Let $g(x) = x(x^2-h^2)$. Setting $g'(x) = 0 = 3x^2 - h^2$, we get the stationary points as $x = \pm h/\sqrt{3}$. The required maximum is

$$\left|x\left(x^2-h^2\right)\right| = \left|\frac{h}{\sqrt{3}}\left(\frac{h^2}{3}-h^2\right)\right| = \frac{2h^3}{3\sqrt{3}}.$$

Therefore, $|Error| \le \dfrac{h^3}{9\sqrt{3}}M_3$.

We have $f(x) = e^{-x}$, $f'(x) = -e^{-x}$, $f''(x) = e^{-x}$, $f'''(x) = -e^{-x}$, and $M_3 = \max_{0\le x\le 1} e^{-x} = 1$.

Choose *h* such that $\dfrac{h^3}{9\sqrt{3}} < 5\times10^{-4}$, or $h^3 < 45\sqrt{3}\times10^{-4}$, or $h < 0.1983$.

Q 6b

```
#include <stdio.h>
# include<conio.h>
float power (float m, int n);
long int fact (int m);
void main ( ) {
        clrscr();
        int i, a;
        float val, temp, x;
        printf ("Input the value of x ");
        scanf("%f", &x);
        fflush(stdin);
```

```
        temp = power(x,2);
        val = 1- (temp/2 );
        a = 2;
        for (i=3; i<=25; i++) {
                temp = (float) fact(a*(i-1));
                val += power (x, (a*(i-1))) / temp;
        }
        printf ("The sum of series is %f ", val );
}
float power (float base, int n) {
        int i;
        float p;
        p = 1;
        for (i=1; i<=n; i++) {
                p = p*base;
        }
        return p;
}
long int fact (int n) {
        int i;
        long int p = 1;
        if (n > 1)
                for (i=2; i<=n; ++i)
                        p = p*i;
        return p ;
}
```

PART II

Q 7a

```
#include <stdio.h>
#include <conio.h>
float T(int,float);
void main() {
        clrscr();
        int n;
        float x,Tn1;
        printf("n = ");
        scanf("%d", &n);
        printf("x = ");
        scanf("%f", &x);
        Tn1 = T(n,x);
        printf("The value of Chebyshev Polynomial = %f\n", Tn1);
}
float T(int a, float x1) {
        int m, k;
        float Tn;
        if (a == 0) {
                return( 1.0);
        }
        else if (a == 1) {
                return(x1);
        }
        else {
                m = a - 1;
                k = a - 2;
                return( 2.0 * x1 * T(m, x1) - T(k, x1));
        }
}
```

Q 7b We have the following divided difference table.

x	$f(x)$	first d.d	second d.d	third d.d
4	48			
5	100	52		
7	294	97	15	
10	900	202	21	1

Newton's divided difference interpolation:

$$f(x) = f(x_0) + (x - x_0) f[x_0, x_1] + (x - x_0)(x - x_1) f[x_0, x_1, x_2]$$
$$+ (x - x_0)(x - x_1)(x - x_2) f[x_0, x_1, x_2, x_3]$$
$$= 48 + (x-4)(52) + (x-4)(x-5)(15) + (x-4)(x-5)(x-7)(1) = x^3 - x^2.$$

Hence, $f(2) = 4$, $f(8) = 448$.

Q 8a Bairstow's formula

$R_p = -c_{n-2}$, $R_q = -c_{n-3}$, $S_p = b_{n-1} - c_{n-1} - pc_{n-2}$, $S_q = -(c_{n-2} + pc_{n-3}),$,

$R = b_{n-1}, S = b_n + pb_{n-1}$, $\Delta p = -\dfrac{RS_q - SR_q}{R_p S_q - R_q S_p}$, $\Delta q = -\dfrac{R_p S - R S_p}{R_p S_q - R_q S_p}$,

$p_1 = p_0 + \Delta p$, $q_1 = q_0 + \Delta q$. $p_0 = 3$, $q_0 = -5$.

	1	5	3	−5	−9
$-p = -3$		−3	−6	−6	3
$-q = 5$			5	10	10
	1	2	2	−1	$4 = b_4$
$-p = -3$		−3	3	−30	
$-q = 5$			5	−5	
	1	−1	10	$-36 = c_3$	

We have $R_p = -10$, $R_q = 1$, $S_p = 5$, $S_q = -7$, $R = -1$, $S = 1$.

$$\Delta p = -\frac{6}{65} = -0.0923, \quad \Delta q = \frac{5}{65} = 0.0769.$$

$p_1 = 3 - 0.0923 = 2.9077$, $q_1 = -5 + 0.0769 = -4.9231$.

The factor is $x^2 + 2.9077x - 4.9231$.

Q 8b Classical Runge-Kutta fourth order method.

$$y_{n+1} = y_n + \frac{1}{6}[k_1 + 2k_2 + 2k_3 + k_4]$$

$$k_1 = h f(x_n, y_n), \quad k_2 = h f\left(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right), \quad k_3 = h f\left(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}\right),$$

$$k_4 = h f(x_n + h, y_n + k_3)$$

We have $f(x, y) = x^2 + y^2$, $x_0 = 1$, $y_0 = 2$, $h = 0.2$.

$k_1 = 0.2(1+4) = 1.0$, $k_2 = 0.2 f(1.1, 2.5) = 1.492$,

$k_3 = 0.2 f(1.1, 2.746) = 1.7501$, $k_4 = 0.2 f(1.2, 3.7501) = 3.1007$.

$$y(1.2) \approx y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) = 3.7642.$$

Q 9a Gauss-Legendre two point formula

$$\int_{-1}^{1} f(x)\,dx = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right).$$

$\int_a^b f(x)\,dx$ is transformed to $\int_{-1}^{1} g(t)\,dt$ by the transformation $x = [(b-a)\,t + (b+a)]/2$.

Transform the interval [5, 12] to [−1, 1]. Use the transformation $x = [7t + 17]/2$.

We have $dx = 7dt/2$ and

$$\int_5^{12} \frac{\mathrm{dx}}{\mathrm{x}} = \int_{-1}^{1} \frac{7\mathrm{dt}}{7\mathrm{t}+17} = 7\left[\frac{1}{\left(-7/\sqrt{3}\right)+17} + \frac{1}{\left(7/\sqrt{3}\right)+17}\right] = 0.8729.$$

Q 9b

```
#include <stdio.h>
#include <conio.h>
void main () {
        clrscr();
        FILE * fp;
        char c;
        int nw=0,ns=0;
        fp = fopen ("sample.txt", "r+" );
        if (fp == NULL)
                printf("Cannot open file.\n");
        else {
                while (( c=fgetc(fp)) != EOF) {
                if ( c == ' ' || c == '\n' || c == '\t' )
                        nw++ ;
                if (  c == '.' )
                        ns++;
                }
                printf ("The total number of words and sentences" );
                printf (" in the file are % d and % d respectively", nw, ns );
        }
        fclose( fp );
}
```

Q 10a $f(x_0, x_1) = \dfrac{f_1 - f_0}{x_1 - x_0} = \dfrac{(1/x_1) - (1/x_0)}{x_1 - x_0} = -\dfrac{1}{x_0 x_1}$

$$f(x_0, x_1, x_2) = \frac{f(x_1, x_2) - f(x_0, x_1)}{x_2 - x_0} = \frac{[-1/(x_1\,x_2)] + [1/(x_0\,x_1)]}{x_2 - x_0} = \frac{(-1)^2}{x_0\,x_1 x_2}.$$

Let the result be true for $n = k$, that is

$$f[x_0, x_1, \cdots, x_k] = \frac{(-1)^k}{x_0 x_1 \cdots x_k}.$$

Then, $$f[x_0, x_1, \cdots, x_{k+1}] = \frac{f[x_1, \cdots, x_{k+1}] - f[x_0, x_1, \cdots, x_k]}{x_{k+1} - x_0}$$

$$= \frac{[(-1)^k/(x_1 x_2 \cdots x_{k+1})] - [(-1)^k/(x_0 x_1 \cdots x_k)]}{x_{k+1} - x_0}$$

$$= \frac{(-1)^{k+1}}{x_0 x_1 \cdots x_{k+1}}.$$

Hence, $$f[x_0, x_1, \cdots, x_n] = \frac{(-1)^n}{x_0 x_1 \cdots x_n}.$$

Q 10 b Unions contain members whose individual data types may differ from one another. However, the members that compose a union all share the same storage area within the computer's memory, whereas

each member within a structure is assigned its own unique storage area. Thus, unions are used to conserve memory. They are useful for applications involving multiple members, where values need not be assigned to all of the members at any one time.

```
union  id {
        char color[12]
        int size;
} shirt, blouse;
```

Here we have two union variables, shirt and blouse, of type id. Each variable can represent either a 12-character string (color) or an integer quantity (size) at any one time.

The 12-character string will require more storage area within the computer's memory than the integer quantity. Therefore, a block of memory large enough for the 12-character string will be allocated to each union variable. The compiler will automatically distinguish between the 12-character array and the integer quantity within the given block of memory, as required.

A structure is a collection of one or more variables, possibly of different types, grouped together under a single name for convenient handling. Structures help to organize complicated data, particularly in large programs, because they permit a group of related variables to be treated as a unit instead of as separate entities. Examples:

(i) The payroll record: an employee is described by a set of attributes such as name, address, social security number, salary etc.

(ii) A point is a pair of co-ordinates, a rectangle is a pair of points and so on.

Q 11a We have the exact solution as $\xi = a^{1/3}$ or $\xi^3 = a$. Set the error as $x_n - \xi = \varepsilon_n$. Substituting in the given equation, we get

$$\xi + \varepsilon_{n+1} = p\left(\xi + \varepsilon_n\right) + \frac{qa}{\xi^2}\left[1 + \frac{\varepsilon_n}{\xi}\right]^{-2}$$

$$= p\left(\xi + \varepsilon_n\right) + q\xi\left[1 - \frac{2\varepsilon_n}{\xi} + 3\frac{\varepsilon_n^2}{\xi^2} + \ldots.\right]$$

$$= (p+q)\xi + (p-2q)\varepsilon_n + \frac{3q\varepsilon_n^2}{\xi} - \ldots.$$

We have two unknowns p and q. Setting $p+q=1,\ \ p-2q=0,$ we get $p=2/3,\ q=1/3$. Hence, for $p=2/3,\ q=1/3$, the method is of order 2, since

$$\varepsilon_{n+1} \approx \frac{3q}{\xi}\varepsilon_n^2 = \frac{1}{\xi}\varepsilon_n^2 \ .$$

The error constant is given by $(1/\xi) = \left(1/a^{1/3}\right)$.

Q 11b Gauss-Seidel method: $\mathbf{x}^{(k+1)} = \mathbf{H}\,\mathbf{x}^{(k)} + \mathbf{c}$

where $\mathbf{H} = -(D+L)^{-1}U$ is the iteration matrix and $A = L + D + U$. The method converges if the spectral radius of $\mathbf{H}$ is less than 1. We have

$$\mathbf{H} = -\begin{bmatrix} 2 & 0 & 0 \\ -1 & 2 & 0 \\ 0 & -1 & 2 \end{bmatrix}^{-1}\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} = -\frac{1}{8}\begin{bmatrix} 4 & 0 & 0 \\ 2 & 4 & 0 \\ 1 & 2 & 4 \end{bmatrix}\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} = -\frac{1}{8}\begin{bmatrix} 0 & -4 & 0 \\ 0 & -2 & -4 \\ 0 & -1 & -2 \end{bmatrix}$$

Eigen values of $\mathbf{H}$: $\left|\mathbf{H} - \lambda\mathbf{I}\right| = 0$.

$$\begin{vmatrix} -\lambda & 1/2 & 0 \\ 0 & (1/4)-\lambda & 1/2 \\ 0 & 1/8 & (1/4)-\lambda \end{vmatrix} = 0 \text{ gives } \lambda\left[\left(\frac{1}{4}-\lambda\right)^2 - \frac{1}{16}\right] = 0, \text{or } \lambda = 0,\ \lambda - \frac{1}{4} = \pm\frac{1}{4};\ \text{or } \lambda = 0, 0, 1/2.$$

We have spectral radius of $\mathbf{H} = \frac{1}{2} < 1$. Hence, the method converges.

Detailed Solutions A-07 DECEMBER 2003

Q1. a. D 0/0 form. Use L' Hospital's rule three times.

.

g. B It is a property of Lagrange fundamental polynomials.

h. A Error equation of Regula–falsi method is $\varepsilon_{k+1} = c\varepsilon_k$.

i. B $\Delta^n\left(e^{ax}\right) = \left(e^{ah} - 1\right)^n e^{ax}$.

j. B

k. B

g. C It is the Gauss–Legendre 3 point formula.

h. D Expand by Taylor series. The first non–vanishing term is $O\left(h^4\right)$.

PART I

Q 2a $\mathbf{A} = LL^T = \begin{bmatrix} 4 & 2 & 3 \\ 2 & 2 & 5 \\ 3 & 5 & 15 \end{bmatrix}$. We get $L = \begin{bmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 3/2 & 7/2 & 1/\sqrt{2} \end{bmatrix}$.

$\mathbf{Ax} = LL^T\mathbf{x} = \mathbf{b}$. Let $L^T\mathbf{x} = \mathbf{z}$. Solve $L\mathbf{z} = \mathbf{b}$. Then, solve $L^T\mathbf{x} = \mathbf{z}$.

By forward substitution of $L\mathbf{z} = \mathbf{b}$, we get $\mathbf{z} = \left[5/2,\ -3/2,\ -1/\sqrt{2}\right]^T$.

By backward substitution of $L^T\mathbf{x} = \mathbf{z}$, we get $\mathbf{x} = \left[1,\ 2,\ -1\right]^T$.

Q 2b Set $e_k = \xi - x_k$. Substitute in the Newton-Raphson method and expand the terms in Taylor series to obtain

$$e_{k+1} = e_k - \frac{[e_k f'(\xi) + (1/2)e_k^2 f''(\xi) + \ldots]}{f'(\xi) + e_k f''(\xi) + \ldots} \qquad \because f(\xi) = 0$$

$$= e_k - [e_k + ce_k^2 + \ldots][1 + 2ce_k + \ldots]^{-1}$$

$$= e_k - [e_k + ce_k^2 + \ldots][1 - 2ce_k + \ldots] = ce_k^2 + \cdots$$

where $c = \frac{1}{2}\frac{f''(\xi)}{f'(\xi)}$ Hence, $e_{k+1} = ce_k^2$. The order of the method is s = 2.

Q 3a We shall use Gauss elimination method and reduce the augmented matrix [**A** | **b**] to [**U** | **z**] where U is an upper triangular matrix.

$$\begin{bmatrix} 1 & 2 & -3 & a \\ 2 & 6 & -11 & b \\ 1 & -2 & 7 & c \end{bmatrix};\ R_2 - 2R_1, R_3 - R_1.\ \begin{bmatrix} 1 & 2 & -3 & a \\ 0 & 2 & -5 & b-2a \\ 0 & -4 & 10 & c-a \end{bmatrix},\ R_3 + 2R_2$$

$$\begin{bmatrix} 1 & 2 & -3 & a \\ 0 & 2 & -5 & b-2a \\ 0 & 0 & 0 & c+2b-5a \end{bmatrix}$$

The system has a solution if $c+2b-5a=0$. Then, the given system has a one parameter family solutions.

Q 3b Let $f(x,y)=3x^2+y^2+2xy-11=0,\quad g(x,y)=x^2+2y^2-5xy+1=0$. New ton's method :

$$\begin{bmatrix}\partial f/\partial x & \partial f/\partial y\\ \partial g/\partial x & \partial g/\partial y\end{bmatrix}_k\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}=-\begin{bmatrix}f\\ g\end{bmatrix}_k$$

Next approximation is $x_{k+1}=x_k+\Delta x,\quad y_{k+1}=y_k+\Delta y$.

We have, $\begin{bmatrix}6x+2y & 2y+2x\\ 2x-5y & 4y-5x\end{bmatrix}_k\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}=-\begin{bmatrix}f\\ g\end{bmatrix}_k$.

We are given that $x_0=0.8,\quad y_0=1.5$. We obtain

$$\begin{bmatrix}7.8 & 4.6\\ -5.9 & 2\end{bmatrix}\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}=-\begin{bmatrix}-4.43\\ 0.14\end{bmatrix}$$

The solution is $\Delta x=0.2224,\quad \Delta y=0.5859$. Hence, $x_1=1.0224,\quad y_1=2.0859$.

Q 4a (i) Value assigned to &a is 1130. Value assigned to &b is 1134. Value assigned to &c is 1138.

(ii) Value of *pa = 0.002. Value of &(*pa) is 1130.

Q 4b Use New ton's divided difference interpolation. Rearrange the data in ascending order of x_i.

x	$f(x)$	first d.d	second d.d	third d.d
0	3			
1	3	0		
3/2	13/4	1/2	1/3	
2	5/3	−19/6	−11/3	−2

New ton's divided difference formula

$$\begin{aligned}f(x)&=f(x_0)+(x-x_0)f[x_0,\ x_1]+(x-x_0)(x-x_1)f[x_0,\ x_1,\ x_2]\\&\quad+(x-x_0)(x-x_1)(x-x_2)f[x_0,\ x_1,\ x_2,\ x_3]\\&=3+(x)(0)+(x)(x-1)\left(\frac{1}{3}\right)+(x)(x-1)\left(x-\frac{3}{2}\right)(-2)=\frac{1}{3}\left(-6x^3+16x^2-10x+9\right)\end{aligned}$$

Q 5a
a = 100, b = 200
count = 1
c = 20 * (1 – 1) = 0
d = 4 * 1 * 1 = 4
funct1(a, c) = 100
funct1(b, d) = 196
count = 2
c = 20 * (2 – 1) = 20
d = 4 * 2 * 2 = 16
funct1(a, c) = 80
funct1(b, d) = 184
count = 3
c = 20 * (3 – 1) = 40
d = 4 * 3 * 3 = 36
funct1(a, c) = 60
funct1(b, d) = 164

count = 4
c = 20 * (4 – 1) = 60
d = 4 * 4 * 4 = 64
funct1(a, c) = 40
funct1(b, d) = 136
count = 5
c = 20 * (5 – 1) = 80
d = 4 * 5 * 5 = 100
funct1(a, c) = 20
funct1(b, d) = 100

Q 5b Substitute $x = x_1 - t$. Then, the given formula becomes

$$\int_{-h}^{h} f(t)dt = \frac{h}{3}[a\,f(h) + b\,f(0) + a\,f(-h)] + ph^2[\,f'(h) - f'(-h)].$$

Make the formula exact for $f(t) = t^i, \quad i = 1, 2, \cdots, 5.$

$f(t) = 1$: $\quad 2h = \frac{h}{3}[2a + b], \quad or \quad 2a + b = 6.$

$f(t) = t$: $\quad 0 = 0,$ satisfied.

$f(t) = t^2$: $\quad \frac{2}{3}h^3 = \frac{h}{3}[2ah^2] + 4ph^3, \quad or \quad a + 6p = 1.$

$f(t) = t^3$: $\quad 0 = 0,$ satisfied.

$f(t) = t^4$: $\quad \frac{2}{5}h^5 = \frac{h}{3}[2ah^4] + 8ph^5, \quad or \quad a + 12p = \frac{3}{5}.$

Solving , we get $a = 7/5, \;\; b = 16/5, \;\; p = -1/15.$

For $f(t) = t^5$, the equation is satisfied. For $f(t) = t^6$, we get

$$c = \left(\frac{2}{7} - \frac{2}{3}a - 12p\right)h^7 = \frac{16}{105}h^7.$$

Error term = $\frac{c}{6!}f^{(6)}(\xi) = \frac{16h^7}{(105)(6!)}f^{(6)}(\xi) = \frac{h^7}{4725}f^{(6)}(\xi), \qquad x_0 < \xi < x_2.$

Q 6a We generate the data as

x	0	1	2	3	4
$f(x)$	1	2	4	8	16

Least squares approximation : $J = \sum\left[f(x_k) - (c + bx_k + ax_k^2)\right]^2 = \text{minimum}.$

Conditions of extremum gives

$$\frac{\partial J}{\partial c} = 0 = \sum\left[f(x_k) - (c + bx_k + ax_k^2)\right], \qquad or \quad N\,c + b\sum x_k + a\sum x_k^2 = \sum f_k$$

$$\frac{\partial J}{\partial b} = 0 = \sum\left[f(x_k) - (c + bx_k + ax_k^2)\right]x_k, \quad or \quad c\sum x_k + b\sum x_k^2 + a\sum x_k^3 = \sum x_k f_k$$

$$\frac{\partial J}{\partial a} = 0 = \sum\left[f(x_k) - (c + bx_k + ax_k^2)\right]x_k^2, \quad or \quad c\sum x_k^2 + b\sum x_k^3 + a\sum x_k^4 = \sum x_k^2 f_k.$$

These are the normal equations. We have $N = 5$,

$\sum x_k = 10, \;\; \sum x_k^2 = 30, \;\; \sum x_k^3 = 100, \;\; \sum x_k^4 = 354, \;\; \sum f_k = 31, \;\; \sum x_k f_k = 98, \;\; \sum x_k^2 f_k = 346.$

Normal equations are

$$\begin{aligned} 5c &+ 10b &+ 30a &= 31 \\ 10c &+ 30b &+ 100a &= 98 \\ 30c &+ 100b &+ 354a &= 346. \end{aligned}$$

The solution is $c = 9/7,\ b = -34/35,\ a = 8/7$. The least squares approximation is

$$f(x) = (45 - 34x + 40x^2)/35 .$$

Q 6b

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
#define d    b * b – 4 * a * c
#define root if (d > 0) {
                    printf("The two real roots are ");
                    printf("%f %f", (–b+sqrt(d))/(2.0*a), (–b–sqrt(d))/(2.0*a));
             }
             else if (d == 0) {
                    printf("The repeated roots are ");
                    printf("%f %f", –b/(2.0*a), –b/(2.0*a));
             }
             else {
                    printf("The complex pair is ");
                    printf("%.2f+%.2fi    ", –b/(2.0*a), sqrt(abs(d))/(2.0*a));
                    printf("and    %.2f–%.2fi", –b/(2.0*a), –sqrt(abs(d))/(2.0*a));
             }

void main() {
   clrscr();
   int a, b, c;
   printf("Enter the value of a, b, c : ");
   scanf("%d %d %d", &a, &b, &c);
   root;
}
```

PART II

Q 7a Partition method :

$$A = \begin{bmatrix} B & C \\ E & D \end{bmatrix}, \quad A^{-1} = \begin{bmatrix} X & Y \\ Z & V \end{bmatrix}$$

where $V = (D - E\,B^{-1}C)^{-1},\ Y = -B^{-1}C\,V,\ Z = -V\,E\,B^{-1},\ X = B^{-1}(I - C\,Z)$.

Let the partition of the given matrix be

$$B = \begin{bmatrix} 1 & 1 \\ 4 & 3 \end{bmatrix}, \quad C = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad E = \begin{bmatrix} 3 & 5 \end{bmatrix}, \quad D = [3].$$

Then, we obtain the following results :

$$B^{-1} = \begin{bmatrix} -3 & 1 \\ 4 & -1 \end{bmatrix}, \quad E\,B^{-1}C = [13], \quad D - E\,B^{-1}C = [-10], \quad V = \begin{bmatrix} -1/10 \end{bmatrix}$$

$$Y = -B^{-1}C\,V = \begin{bmatrix} -2/5 & 1/2 \end{bmatrix}^T, \quad Z = -V\,E\,B^{-1} = \begin{bmatrix} 11/10 & -1/5 \end{bmatrix}, \quad X = B^{-1}(I - C\,Z) = \begin{bmatrix} 14/10 & 1/5 \\ -15/10 & 0 \end{bmatrix}$$

$$A^{-1} = \begin{bmatrix} X & Y \\ Z & V \end{bmatrix} = \begin{bmatrix} 14/10 & 1/5 & -2/5 \\ -15/10 & 0 & 1/2 \\ 11/10 & -1/5 & -1/10 \end{bmatrix}$$

Q 7b Power method :

$$\mathbf{y}_{k+1} = \mathbf{A}\mathbf{v}_k, \qquad \mathbf{v}_{k+1} = \mathbf{y}_{k+1}/m_{k+1}, \qquad m_{k+1} = \max_r \left|(\mathbf{y}_{k+1})_r\right|$$

$$\lambda = \lim_{k \to \infty} \left[\frac{(\mathbf{y}_{k+1})_r}{(\mathbf{v}_k)_r} \right], \quad r = 1, 2, \ldots, n.$$

All n ratios tend to the largest eigen value in magnitude.

$\mathbf{v}_0 = [1,\ 1,\ 1]^T$, $\mathbf{y}_1 = \mathbf{A}\mathbf{v}_0 = [-8,\ 4,\ 18]^T$, $\mathbf{v}_1 = [-4/9,\ 2/9,\ 1]^T$,

$\mathbf{y}_2 = \mathbf{A}\mathbf{v}_1 = [95/9,\ -10/9,\ -70/9]^T$.

After two iterations, the ratios for $|\lambda|$ are 95/4, 5, 70/9. The ratios have not yet converged.

Q 8a Jacobi method : Let $|a_{ik}|$ be the largest off-diagonal element in magnitude. Choose θ such that

$$\tan 2\theta = \frac{2a_{ik}}{a_{ii} - a_{kk}}, \quad \text{or} \quad \theta = \frac{1}{2}\tan^{-1}\left(\frac{2a_{ik}}{a_{ii} - a_{kk}}\right).$$

If $a_{ii} = a_{kk}$, then $\theta = \pi/4$, if $a_{ik} > 0$; and $-\pi/4$, if $a_{ik} < 0$. Then, the elements a_{ii}, a_{ik}, a_{ki}, a_{kk} form the 2×2 rotation sub-matrix. Repeat the procedure until *A* is reduced to a diagonal matrix. Then, the eigen values are on the diagonal of this matrix. The product of the rotation matrices $S = S_1 S_2 \ldots S_r$ gives the matrix of eigenvectors. The first column of *S* is the eigenvector corresponding to the eigen value in the first (1×1) location of the diagonal matrix, etc.

First rotation : $a_{12} = a_{21} = 3$ is the largest off-diagonal element.

$$\theta = \frac{1}{2}\tan^{-1}\left(\frac{6}{2-2}\right) = \frac{\pi}{4}.$$

$$S_1 = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \qquad A_1 = S_1^{-1} A S_1 = S_1^T A S_1 = \begin{bmatrix} 5 & 0 & 3/\sqrt{2} \\ 0 & -1 & 1/\sqrt{2} \\ 3/\sqrt{2} & 1/\sqrt{2} & 1 \end{bmatrix}$$

Second rotation : $a_{13} = a_{31} = 3/\sqrt{2}$ is the largest off-diagonal element.

$\theta = \frac{1}{2}\tan^{-1}\left(\frac{3\sqrt{2}}{5-1}\right)$, or $\theta = 0.40741346$ radians, $\sin\theta = 0.396236$, $\cos\theta = 0.918149$.

$$S_2 = \begin{pmatrix} 0.918149 & 0 & -0.396236 \\ 0 & 1 & 0 \\ 0.396236 & 0 & 0.918149 \end{pmatrix}.$$

$$A_2 = S_2^{-1} A_1 S_2 = S_2^T A_1 S_2 = \begin{bmatrix} 5.915479 & 0.280181 & 0.0 \\ 0.280181 & -1 & 0.649229 \\ 0.0 & 0.649229 & 0.084525 \end{bmatrix}$$

$$S = S_1 S_2 = \begin{bmatrix} 0.64923 & -0.70711 & -0.280181 \\ 0.64923 & 0.70711 & -0.280181 \\ 0.396236 & 0 & 0.918149 \end{bmatrix}.$$

Iterations have not converged since *A* is not reduced as yet as a diagonal matrix. The eigen values may be near 5.9, −1, 0.08. The columns of *S* are the corresponding eigenvectors.

Q 8b

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>
char** swap(int, char *p1[]);
void main() {
        clrscr();
        char *p[10];
```

```
        int i=0,n,j;
        printf("Enter no of strings : ");
        scanf("%d", &n);
        for (i=0; i<n; i++) {
                p[i] = (char *) malloc(2*n);
                printf("Enter string : ");
                scanf("%s",p[i]);
                fflush(stdin);
        }
        char **q = swap (n, p);
        printf("The ascending order of strings is :\n");
        for (i=0; i<n; i++)
                printf("%d : %s\n",i+1, *(q+i));
}
char** swap(int n1, char *p1[]) {
        char *temp;
        int i,j;
        for (i=0; i<n1; i++) {
                for (j=i+1; j<n1; j++) {
                        if (strcmp(*(p1+i), *(p1+j)) > 0) {
                                temp = *(p1+i);
                                *(p1+i) = *(p1+j);
                                *(p1+j) = temp;
                        }
                }
        }
        return p1;
}
```

Q 9a The bound on the error in linear interpolation is given by

$$\left|\text{Errror}\right| = \frac{1}{2}\left|(x-x_0)(x-x_1)\right|\left|f''(\xi)\right| \le \frac{1}{2}\max_{[x_0,x_1]}\left|(x-x_0)(x-x_1)\right| \max_{[x_0,x_1]}\left|f''(x)\right|.$$

Now, the stationary point of $(x-x_0)(x-x_1)$ is $x=(x_0+x_1)/2$. Hence,

$$|\text{ Error }| \le \frac{1}{8}(x_1-x_0)^2 M_2 = \frac{h^2}{8}M_2,$$

where $h = x_1 - x_0$ and $M_2 = \max_{[x_0,x_1]}\left|f''(x)\right|$.

We have $f(x)=e^x,\ f'(x)=e^x,\ f''(x)=e^x$ and $M_2 = \max_{[0,1]}[e^x] = e$.

We require $\frac{h^2}{8}e \le 5\times 10^{-4}$, or $h \le \left[\frac{40}{e}\times 10^{-4}\right]^{1/2} = 0.03836.$

Q 9b (i) Make the formula exact for $f(x) = 1, x, x^2$.

$f(x)=1$: $\quad 1 = a+b+c$

$f(x)=x$: $\quad \frac{1}{2} = \frac{b}{2}+c$

$f(x)=x^2$: $\quad \frac{1}{3} = \frac{b}{4}+c.$

The solution of this system is $a=1/6,\ b=2/3,\ c=1/6$. The formula is given by

$$\int_0^1 f(x)\,dx = \frac{1}{6}\left[f(0)+4f\left(\frac{1}{2}\right)+f(1)\right].$$

(ii) Make the formula exact for $f(x) = 1, x, x^2$.

$f(x)=1$: $\quad 1=\alpha+\beta+\gamma$

$f(x)=x$: $\quad \frac{1}{2}=\frac{\alpha}{4}+\frac{\beta}{2}+\frac{3\gamma}{4}$

$f(x)=x^2$: $\quad \frac{1}{3}=\frac{\alpha}{16}+\frac{\beta}{4}+\frac{9\gamma}{16}.$

The solution of this system is $\alpha=2/3,\ \beta=-1/3,\ \gamma=2/3.$ The formula is given by

$$\int_0^1 f(x)\,dx=\frac{1}{3}\left[2f\left(\frac{1}{4}\right)-f\left(\frac{1}{2}\right)+2f\left(\frac{3}{4}\right)\right].$$

Both the methods are exact for $f(x)=x^3$.

For $f(x)=x^4$, the error constant for the first method is $-1/120$ and for the second method is $7/960$. Hence the second method is better since it has a smaller error constant.

Q 10a (i)

```
#include <stdio.h>
void main() {
        int i, sum=0;
        for (i=2; i<100; i+=3)
                if (i%5 == 0)
                        if ((i/5)%2 ==0)
                                sum += i;
        printf("The sum is %d", sum);
}
```

Q 10a (ii)

```
#include <stdio.h>
void main() {
        int i, sum=0;
        for (i=2; i<100; i+=3)
                sum += (i%5 == 0) ? (((i/5)%2 == 0) ? i : 0) : 0;
        printf("The sum is %d", sum);
}
```

Q 10b Given system is $\mathbf{A}\,\mathbf{x}=\mathbf{b}$ and Gauss-Seidel method is $\mathbf{x}^{(k+1)}=H\,\mathbf{x}^{(k)}+\mathbf{c}, k=0,1,\ldots$

where, iteration matrix $=H=-(D+L)^{-1}U$ and $\mathbf{c}=(D+L)^{-1}b$.

$$H=-\begin{bmatrix}2&0&0\\1&6&0\\4&-3&8\end{bmatrix}^{-1}\begin{bmatrix}0&-1&0\\0&0&-2\\0&0&0\end{bmatrix}=-\frac{1}{96}\begin{bmatrix}48&0&0\\-8&16&0\\-27&6&12\end{bmatrix}\begin{bmatrix}0&-1&0\\0&0&-2\\0&0&0\end{bmatrix}$$

$$=-\frac{1}{96}\begin{bmatrix}0&-48&0\\0&8&-32\\0&27&-12\end{bmatrix}$$

Eigen values of H: $|H-\lambda I|=0$ gives the characteristic equation as

$$\lambda(24\lambda^2-\lambda+2)=0.$$

Hence, $\lambda=0$, $\lambda=\frac{1\pm\sqrt{191}\,i}{48}$, and $|\lambda|=0.2887$.

Spectral radius of $H=\rho(H)=0.2887<1$.

Hence, the iteration converges.

Q 11a Runge-Kutta classical fourth order method

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1 = h\,f(x_n,\ y_n),\ \ k_2 = h\,f\left(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right),\ \ k_3 = h\,f\left(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}\right),$$

$$k_4 = h\,f(x_n + h,\ y_n + k_3)\ .$$

We have $f(x, y) = x + y,\ \ x_0 = 1,\ \ y_0 = 2,\ \ h = 0.1.$

$n = 0$: $k_1 = 0.1\,f(1, 2) = 0.3,\ \ k_2 = 0.1\,f(1.05, 2.15) = 0.32,$

$k_3 = 0.1\,f(1.05, 2.16) = 0.321,\ \ k_4 = 0.1\,f(1.1,\ 2.321) = 0.3421,$

$y(1.1) \approx y_1 = 2.3207.$

$n = 1$: $x_1 = 1.1,\ \ y_1 = 2.3207.$

$k_1 = 0.1\,f(1.1, 2.3207) = 0.3421,\ \ k_2 = 0.1\,f(1.15, 2.4918) = 0.3642,$

$k_3 = 0.1\,f(1.15, 2.5028) = 0.3653,\ \ k_4 = 0.1\,f(1.2, 2.686) = 0.3886.$

$y(1.2) \approx y_2 = 2.6857.$

Q 11b The initial value problem (*I V P*) is $y' = f(x, y),\ y(x_0) = y_0$.

Existence theorem : Let $f(x, y)$ be continuous and bounded at all points in some closed rectangular region about (x_0, y_0) , that is, in $|x - x_0| \le a,\ \ |y - y_0| \le b.$ Then, the *I V P* has at least one solution.

Now, $f(x, y) = (x + \sin y)^2$ is continuous in every rectangle about $x_0 = 0,\ y_0 = 3.$

Also, $|x + \sin y|^2 \le 4$ for $-1 \le x \le 1$ and all y. Hence, the given *I V P* has at least one solution.

Uniqueness theorem: Let the *I V P* satisfy the existence theorem and let $f(x, y)$ satisfy the Lipschitz condition

$$|f(x, u) - f(x, v)| \le L|u - v|\ .$$

Then, the solution of the *I V P* is unique. A sufficient condition is $\partial f/\partial y$ should be bounded, that is, $|\partial f/\partial y| \le M$ in the considered domain.

Now, $\dfrac{\partial f}{\partial y} = 2(x + \sin y)\cos y$, and $\left|\dfrac{\partial f}{\partial y}\right| \le 4$ for $x \in [-1, 1]$ and for all y . Hence, the solution is unique.

Detailed Solution A-07 June 2004

Q1. a. **C** Set $f(x) = x^3$. Then, $C = \int_{-1}^{1} x^3 dx - \frac{1}{2}\left[-1 + 3\left(\frac{1}{27}\right)\right] = \frac{4}{9}$.

l. **A** $Lim\ x_{k+1} = x = Lim\ x_k$. This gives $x^3 = q$, or $x = q^{1/3}$.

m. **D** Romberg formula: $I = \{h_2^2 I(h_1) - h_1^2 I(h_2)\}/(h_2^2 - h_1^2)$, with $h_2 = 1/4$, $h_1 = 1/3$. We obtain the result as 0.5867.

n. **B** $|\text{Error}(x)| \le 0.5|(x-0.1)(x-0.2)| M_2$. $\therefore |\text{Error}(0.15)| \le 0.00125 M_2$

o. **C** $N = 4$. Normal equations are $4a + 10b = 50$, $10a + 30b = 150$. The solution is $a = 0$, $b = 5$. $y = 5x$.

p. **A** Iteration matrix = $H = \begin{bmatrix} 0 & 4 \\ 4 & 0 \end{bmatrix}$. Spectral radius of $H = 4 > 1$. *D*iverges.

q. **D**

r. **B**

PART I

Q 2a $f(0) = 1$, $f(-1) = -6$. Root lies in (-1, 0)

$$x_{k+1} = x_k - \frac{f_k}{f_k'} = x_k - \frac{7x_k^3 + 8x_k^2 + 8x_k + 1}{21x_k^2 + 16x_k + 8}.$$

We get $x_0 = -0.5$, $x_1 = -0.14286$, $x_2 = -0.14286$. Hence, $x \approx -0.143$.

Q 2b

```
#include <stdio.h>
#include <math.h>
main()
{
        float xinitial, eps, fx, dfx, xnew;
        int i, n;
        float f(float s);
        float df(float s);
        printf("Input value initial approximation xinitial\n");
        printf("n: number of iterations\n");
        printf("eps: error tolerance\n");
        scanf("%f %d %e", &xinitial, &n, &eps);
        printf("xinitial = %f  n = %d  eps = %e\n\n", xinitial, n, eps);
        /* Calculate f and its first derivative at xinitial     */
        for(i = 1;  i <= n; i++)
        {
                fx = f(xinitial);
                dfx = df(xinitial);
                xnew = xinitial – fx / dfx;
                fx = f(xnew);
                if(fabs(fx) <= eps) goto l10;
                /*  Iteration is stopped when abs(f(x)) is less than or equal to eps.
                Alternate conditions can also be used.          */
                xinitial = xnew;
        }
        printf("\nITERATIONS ARE NOT SUFFICIENT");
        goto l20;

l10:    printf("Iterations = %d", i);
printf("  Root = %10.7f,  f(x) = %e\n", xnew, fx);
```

```
l20:    return 0;
}
float f(float x)
{
        float fun;
        fun = 7.0*x*x*x+8*x*x+8.0*x+1;
        return(fun);
}
float df(float x)
{
        float dfun;
        dfun = 21.0*x*x+16.0*x+8.0;
        return(dfun);
}
```

Q 3a Bairstow's formula

$$R_p = -c_{n-2},\ R_q = -c_{n-3},\quad S_p = b_{n-1} - c_{n-1} - pc_{n-2},\quad S_q = -(c_{n-2} + pc_{n-3}),,$$

$$R = b_{n-1}, S = b_n + pb_{n-1},\qquad \Delta p = -\frac{RS_q - SR_q}{R_pS_q - R_qS_p},\qquad \Delta q = -\frac{R_pS - RS_p}{R_pS_q - R_qS_p},$$

$$p_1 = p_0 + \Delta p,\quad q_1 = q_0 + \Delta q.\quad p_0 = 0.4,\quad q_0 = 1.1.$$

	1	4.5	3	4
$-p = -0.4$		−0.4	−1.64	−0.104
$-q = -1.1$			−1.1	−4.51
	1	4.1	0.26	−0.614
$-p = -0.4$		−0.4	−1.48	
$-q = -1.1$			−1.1	
	1	3.7	−2.32	

$$\Delta p = 0.0969,\qquad \Delta q = -0.0984,\qquad p_1 = 0.4969,\qquad q_1 = 1.0016.$$

Q 3b $A^{-1} = (U\,U^T)^{-1} = (U^{-1})^T U^{-1},\qquad U = \begin{pmatrix} \sqrt{3/2} & 3/2 & 1/2 \\ 0 & 1 & -1 \\ 0 & 0 & 2 \end{pmatrix};\qquad U^{-1} = \frac{1}{\sqrt{6}}\begin{pmatrix} 2 & -3 & -2 \\ 0 & \sqrt{6} & \sqrt{3/2} \\ 0 & 0 & \sqrt{3/2} \end{pmatrix}$

$$A^{-1} = (U^{-1})^T U^{-1} = \frac{1}{6}\begin{pmatrix} 4 & -6 & -4 \\ -6 & 15 & 9 \\ -4 & 9 & 7 \end{pmatrix}.$$

Q 4a $x_1^{(k+1)} = \frac{1}{6}\left[9 - 3x_2^{(k)} - x_3^{(k)}\right],\qquad x_2^{(k+1)} = \frac{1}{5}\left[5 + 2x_1^{(k+1)} + 2x_3^{(k)}\right],$

$$x_3^{(k+1)} = \frac{1}{8}\left[-4 - 3x_1^{(k+1)} - 2x_2^{(k+1)}\right].$$

$$\mathbf{x}^{(0)} = [1\quad 1\quad -1]^T,\quad \mathbf{x}^{(1)} = [1.1667\quad 1.0667\quad -1.2042],\qquad \mathbf{x}^{(2)} = [1.1674\quad 0.9853\quad -1.1841].$$

Iteration matrix $= H = -(D+L)^{-1}U = \frac{1}{240}\begin{pmatrix} 0 & -120 & -40 \\ 0 & -48 & 80 \\ 0 & 57 & -5 \end{pmatrix}$.

Characteristic equation of H : $\lambda\,(240\lambda^2 - 53\lambda - 18) = 0.$

Eigen values of H : $\lambda = 0,\quad \frac{53 \pm 141.7357}{480} = 0.4057,\quad -0.1849$.

Spectral radius of H : $\rho(H) = 0.4057.$

Rate of convergence $= -\log_{10}\rho(H) = 0.3918$ *or as* $= -\ln\rho(H) = 0.902.$

Q 4b The Power method is given as the following

$\mathbf{y}_{k+1} = A\,\mathbf{v}_k,\quad \mathbf{v}_{k+1} = \mathbf{y}_{k+1}/m_{k+1},\quad m_{k+1} = \max_r \left|(\mathbf{y}_{k+1})_r\right|,\quad \lambda = \left|(\mathbf{y}_{k+1})_r/(\mathbf{v}_k)_r\right|.$

$\mathbf{v}_0$ is given as $[1,\ 1,\ 1]^T$.

$\mathbf{y}_1 = [1,\ 17,\ -1]^T$, $\mathbf{v}_1 = [0.05882,\ 1,\ -0.05882]^T$, $\mathbf{y}_2 = [-0.88236,\ 20.05882,\ -2.05882]^T$;

$\mathbf{v}_2 = [-0.04399,\ 1,\ -0.10264]^T$, $\mathbf{y}_3 = [-1.08798,\ 20.24927,\ -2.10264]^T$,

$\mathbf{v}_3 = [-0.05373,\ 1,\ -0.10384]^T$, $\mathbf{y}_4 = [-1.1075,\ 20.26141,\ -2.10384]^T$,

$\mathbf{v}_4 = [-0.05466,\ 1,\ -0.10384]^T$, $\mathbf{y}_5 = [-1.10932,\ 20.26233,\ -2.10384]^T$,

$\mathbf{v}_5 = [-0.05474,\ 1,\ -0.10383]^T$.

Ratios of $|\lambda|$: 20.295, 20.262, 20.261, $|\lambda| \approx 20.3$ and the corresponding eigen vector is v_5.

Q 5a Substitute $x_k = \xi + \varepsilon_k$ in the secant method to obtain

$$\varepsilon_{k+1} = \varepsilon_k - \frac{(\varepsilon_k - \varepsilon_{k-1})f(\xi+\varepsilon_k)}{f(\xi+\varepsilon_k) - f(\xi+\varepsilon_{k-1})}$$

$$= \varepsilon_k - \frac{(\varepsilon_k - \varepsilon_{k-1})[\varepsilon_k f'(\xi) + (\varepsilon_k^2/2)f''(\xi) + \cdots]}{(\varepsilon_k - \varepsilon_{k-1})f'(\xi) + (1/2)(\varepsilon_k^2 - \varepsilon_{k-1}^2)f''(\xi) + \cdots}$$

$$= \varepsilon_k - [\varepsilon_k + c\varepsilon_k^2 + \cdots][1 + (\varepsilon_{k-1} + \varepsilon_k)c + \cdots]^{-1}$$

$$= \varepsilon_k - [\varepsilon_k + c\varepsilon_k^2 + \cdots][1 - (\varepsilon_{k-1} + \varepsilon_k)c + \cdots]$$

$$= \varepsilon_k - [\varepsilon_k - c\varepsilon_k\varepsilon_{k-1}] + \cdots = c\varepsilon_k\varepsilon_{k-1} + \cdots$$

where $c = \frac{1}{2}\frac{f''(\xi)}{f'(\xi)}$.

Hence, we have $\varepsilon_{k+1} = c\ \varepsilon_k\ \varepsilon_{k-1}$.

$\varepsilon_{k+1} = A\,\varepsilon_k^p$ gives $\varepsilon_k = A\,\varepsilon_{k-1}^p$, or $\varepsilon_{k-1} = \varepsilon_k^{(1/p)}A^{-(1/p)}$. Hence,

$\varepsilon_{k+1} = A\,\varepsilon_k^p = cA^{-(1/p)}\varepsilon_k^{1+(1/p)}$. Comparing the powers of ε_k, we get

$p = 1 + \frac{1}{p}$, or $p^2 - p - 1 = 0$, $p = \frac{1}{2}\left(1 \pm \sqrt{5}\right)$. Since, $p > 0$, we get $p = 1.618 = 1.62$.

Q 5b

```
# include <stdio.h>
void main ( )
{
        int years,method,i;
        float depreciation,value,original;
        printf  ( "Method used : ( 1 ) Straight line.. ( 2 ) Sum.." );
        scanf  ("%d", &method );
```

```
                printf  ( "Original value: " );
                scanf  ( " %f  ",  &original );
                printf  ( "No. of  years : " );
                scanf  ( " %d " , &years );
                switch (method)
                {
                case 1 :
                        printf  ( " \ n  Straight line method \ n " );
                        depreciation = original / years;
                        for ( i =1; i <= years; ++i )
                        {
                                original -= depreciation ;
                                printf  ( "End of  year % 2d ", i );
                                printf  ( " Depreciation : % f  ", depreciation );
                                printf  ( " Current value : % f  \ n ", original );
                        }
                        break ;
                case 2 :
                        printf  ( " \ n Sum of year's digits method \  n " );
                        value = original ;
                        for ( i = 1 ; i <= years; ++i )
                        {
                                depreciation = ( years - i + 1 ) * original / ( years*(years+1)/2);
                                value -=depreciation;
                                printf  ( " end of  year  % 2d ", i );
                                printf  ( "depreciation : % f  ", depreciation );
                                printf  ( " current value : % f  \ n ", value );
                        }
                        break ;
                }
        }
```

Q 6a Inverse power method:

$\mathbf{y}_{k+1} = A^{-1}\,\mathbf{v}_k$, $\mathbf{v}_{k+1} = \mathbf{y}_{k+1}/m_{k+1}$, $|\mu| = \underset{k\to\infty}{Lim}\left|\dfrac{(\mathbf{y}_{k+1})_r}{(\mathbf{v}_k)_r}\right|$ $r = 1, 2, \ldots, n$

$\mathbf{v}_{k+1}$ is the required eigen vector . $|\lambda| = 1/|\mu|$.

$$A^{-1} = \begin{bmatrix} 4 & -1 & -1 \\ 2 & -1 & 0 \\ -5 & 2 & 1 \end{bmatrix}. \qquad \mathbf{v}_0 = [-0.8 \quad -0.3 \quad 1.0]^T$$

$\mathbf{y}_1 = [-3.9 \quad -1.3 \quad 4.4]^T$, $\mathbf{v}_1 = [-0.8864 \quad -0.2955 \quad 1]^T$, $\mathbf{y}_2 = [-4.2501 \quad -1.4773 \quad 4.841]^T$,
$\mathbf{v}_2 = [-0.8779 \quad -0.3052 \quad 1]^T$, $\mathbf{y}_3 = [-4.2064 \quad -1.4506 \quad 4.7791]^T$, $\mathbf{v}_3 = [-0.8802 \quad -0.3035 \quad 1]^T$,
$\mathbf{y}_4 = [-4.2173 \quad -1.4569 \quad 4.794]^T$, $\mathbf{v}_4 = [-0.8797 \quad -0.3039 \quad 1]^T$, $\mathbf{y}_5 = [-4.2149 \quad -1.4555 \quad 4.7907]^T$.

$|\text{Ratios}|$: 4.791, 4.789, 4.791 . $\therefore |\mu| = 4.79$, $|\lambda| = \dfrac{1}{|\mu|} = \dfrac{1}{4.79} \approx 0.2088$

Q 6b $\tan\theta = \dfrac{a_{13}}{a_{12}} = -1$, $\theta = -\dfrac{\pi}{4}$, $S = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/\sqrt{2} & 1/\sqrt{2} \\ 0 & -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix}$.

$$B = S^{-1}A\,S = S^{T}AS = \begin{pmatrix} 1 & -4/\sqrt{2} & 0 \\ -4/\sqrt{2} & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix}.$$

Sturm sequence: Define $f_n = |\lambda I - B|$. Then

$f_0 = 1, \quad f_1 = \lambda - b_1, \; f_r = (\lambda - b_r) f_{r-1} - c_{r-1}^2 f_{r-2}, \quad r = 2, \cdots, n..$
$f_0 = 1, \quad f_1 = \lambda - 1, \quad f_2 = \lambda f_1 - 8 f_0, \quad f_3 = (\lambda - 2) f_2.$

Form the Sturm table

λ	f_0	f_1	f_2	f_3	V(λ)
–3	+	–	+	–	3
–2	+	–	–	+	2
–1	+	–	–	+	2
0	+	–	–	+	2
1	+	+	–	+	2
2	+	+	–	0	eigenvalue
3	+	+	–	–	1
4	+	+	+	+	0

From the Sturm table, we find that there are eigen values in (– 3, –2), (3, 4) and 2 is an eigen value.

$\therefore$ | Largest eigen value | lies in (3, 4).

PART II

Q 7a Error in quadratic interpolation is bounded by $|\text{error}| \le \frac{h^3}{9\sqrt{3}} M_3, \quad M_3 = \max_{(a,b)} |f'''(x)|$

We have $f''' = 192(3 + 2x)$ and $M_3 = 960$ for $x \in [0, 1]$.

$\frac{h^3}{9\sqrt{3}} 960 < 10^{-6}$ gives $h < 0.00253.$

Q 7b

x	$f(x)$	Δf	$\Delta^2 f$	$\Delta^3 f$
–1	1.2			
0	1.0	–0.2		
1	3.8	2.8	3.0	
2	9.6	5.8	3.0	0
3	18.4	8.8	3.0	0
4	30.2	11.8	3.0	0

We have $h = 1.0$.

$$f(x) = f(x_0) + (x - x_0)\frac{\Delta f_0}{1!h} + \frac{(x - x_0)(x - x_1)}{2!h^2}\Delta^2 f_0$$

$$= 1.2 - 0.2(x + 1) + (x + 1)x(1.5) = 1.5x^2 + 1.3x + 1 \qquad \therefore f(0.5) = 2.025.$$

Q 8a $I = \frac{h}{3}[f_0 + 4(f_1 + f_3 + \cdots + f_{2N-2}) + 2(f_2 + f_4 + \cdots + f_{2N-1}) + f_{2N}].$

We have $f(x) = 1/(x^2 + 2x + 10).$

$h = 1.0;\ \ I = \frac{1}{3}[f(0) + 4f(1) + f(2)] = 0.15442.$

$h = 0.5;\ \ I = \frac{1}{6}[f(0) + 4\{f(0.5) + f(1.5)\} + 2f(1) + f(2)] = 0.15454.$

Romberg value = $\dfrac{16\,I(0.5) - I(1.0)}{15} = 0.15455.$

Q 8b

```
#include <stdio.h>
#include <math.h>
float f();
main()
{
        float   a, b, h, sum, x, trap;
        int     n, i, m;
        printf("Input limits a & b and no. of subintervals n\n");
        scanf("%f %f %d", &a, &b, &n);
        printf("Limits are a = %f, b = %f\n", a, b);
        printf("Number of subintervals = %d\n", n);
        h = (b − a) / n;
        sum = 0.0;
        m = n − 1;
        for (i = 1;  i <= m; i++)
                {
                        x = a + i * h;
                        sum = sum + f(x);
                }
        trap = h * (f(a) + 2.0 * sum + f(b)) / 2.0;
        printf("Value of integral with %d ", n);
        printf("Subintervals = %14.6e\n", trap);
        return 0;
}
float   f(float x)
{
        float    fun;
        fun = 1.0 / (10.0 + 2.0 * x + x * x);
        return(fun);
}
```

Q 9a $I = \sum\left[y_k - \left(a + \dfrac{b}{x_k}\right)\right]^2 =$ minimum .

Normal equations : $a\,N + b\sum\frac{1}{x_k} = \sum y_k\ ,\qquad a\sum\frac{1}{x_k} + b\sum\frac{1}{x_k^2} = \sum\frac{y_k}{x_k}$

We have N = 6, $\sum(1/x_k) = 21,\ \sum(1/x_k^2) = 136.5,\ \sum y_k = 51,\ \sum(y_k/x_k) = 295.5.$

Normal equations are $6a + 21b = 51,\qquad 21a + 136.5b = 295.5.$

Solution is $a = 2,\ b = 1.8571.\qquad \therefore\ p(x) = 2 + (1.8571/x).$

Q 9b $P(x) = \sum_{i=0}^{n} A_i(x) f(x_i) + \sum_{i=0}^{n} B_i(x) f'(x_i).$

$A_i(x) = [1 - 2(x - x_i)\,l_i'(x_i)]\ l_i^2(x),\qquad B_i(x) = (x - x_i)\ l_i^2(x)$

$l_0(x) = -(x-2), \quad l_0' = -1, \quad l_1(x) = x-1, \quad l_1'(x) = 1$

$A_0(x) = \left[1 + 2(x-1)\right](x-2)^2, \quad A_1(x) = \left[1 - 2(x-2)\right](x-1)^2$

$B_0(x) = (x-1)(x-2)^2, \quad B_1(x) = (x-2)(x-1)^2$

$P(x) = (2x-1)(x-2)^2(9) + (5-2x)(x-1)^2(87) + (x-1)(x-2)^2(23) + (x-2)(x-1)^2(163)$

$= 30x^3 - 65x^2 + 63x - 19.$

$\therefore$ P(1.5) = 30.5

Q 10a Choose three points as $x_0 = 0, \quad x_1, \quad x_2 = 1.$

Define $\varepsilon(x) = (x^2 + 5x + 3) - (a + bx).$

We have $\varepsilon(0) + \varepsilon(x_1) = 0, \quad \varepsilon(x_1) + \varepsilon(1) = 0, \quad \varepsilon'(x_1) = 0.$ Hence, we get

$(3-a) + \left[x_1^2 + 5x_1 + 3 - (a + bx_1)\right] = 0.$

$\left[x_1^2 + 5x_1 + 3 - (a + bx_1)\right] + \left[9 - (a+b)\right] = 0,$ and $\quad 2x_1 + 5 - b = 0.$

The solution of this system is $b = 6, \; x_1 = 1/2, \; a = 23/8.$ Hence, $\quad p(x) = (23/8) + 6x.$

Q 10b

```
# include <stdio.h>
long  fibonacci ( int count );
main ( )
{
     int count, n;
     printf( " Number of fibonacci numbers : " );
     scanf(" %d ",  &n);
     printf  ( " \n" );
     for ( count = 1 ; count <= n ; ++count )
          printf  ( " \n i = %2d , F = %1d", count , fibonacci(count) );
}
long fibonacci ( int count )
          /* F = 1 for i < 3 , and F = F1 + F2 for  i > = 3 */
{
          static long int   f1 = 1, f2 =1 ;
           long int  f ;
           f = ( count < 3 ) ? 1 : f1 + f2 ;
           f2 = f1 ;
           f1 = f ;
           return f;
}
```

Q 11a $y' = 2x^2 + 3y^2; \quad f(x,y) = 2x^2 + 3y^2$

$x_0 = 1, \quad y_0 = 1, \qquad y(x_n + h) = y(x_n) + h\, y'(x_n) + \frac{h^2}{2} y''(x_n), \quad y'' = 4x + 6yy'.$

$h = 0.1, \quad y(1.1) = y(1) + 0.1\, y'(1) + \frac{0.01}{2} y''(1) = 1 + 0.1(5) + 0.005(34) = 1.67.$

$h = 0.1, \quad x_1 = 1.1, \quad y_1 = 1.67.$

$y(1.2) = y(1.1) + 0.1\, y'(1.1) + 0.005\, y''(1.1)$

$= 1.67 + 0.1(10.7867) + 0.005(112.4827) = 3.3111.$

Q 11b $y' = x^2 + y^2, \quad y(0) = 1$

$f(x, y) = x^2 + y^2, \quad x_0 = 0, \quad y_0 = 1$

$k_1 = h\, f(x_n,\ y_n), \quad k_2 = h\, f\left(x_n + \frac{h}{2},\ y_n + \frac{k_1}{2}\right),$

$k_3 = h\, f\left(x_n + \frac{h}{2},\ y_n + \frac{k_2}{2}\right), \quad k_4 = h\, f(x_n + h,\ y_n + k_3),$

$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4).$

$.h = 0.2, \quad x_0 = 0, \quad y_0 = 1.$

$k_1 = 0.2, \ k_2 = 0.244, \ k_3 = 0.2538, \ k_4 = 0.3224, \quad y(0.2) = 1.253.$

$x_1 = 0.2, \quad y_1 = 1.253.$

$k_1 = 0.3220, k_2 = 0.4179, k_3 = 0.4455, k_4 = 0.6090, \quad y(0.4) = 1.6960.$

Detailed Solutions A-07 December 2004

Q1. a. **B** Relative error = $|\text{exact value} - \text{numerical value}|/|\text{exact value}| = 0.0016$.

s. **A** Number of iterations required $\geq$ [log(b – a) – log ε]/ log2 = – log ε /log2.

t. **D** Iteration matrix = $H = \begin{bmatrix} 0 & -p \\ 0 & -p^2 \end{bmatrix}$. Spectral radius of $H = p^2$. Convergence if $|p| < 1$.

u. **C** $f[x_1, x_2] = x_1^2 + x_1x_2 + x_2^2$, $f[x_2, x_3] = x_2^2 + x_2x_3 + x_3^2$, $f[x_1, x_2, x_3] = x_1 + x_2 + x_3$.

v. **B** Least squares error = $\sum [f(x_k) - 5x_k]^2 = 4$.

w. **A** Use Taylor series expansions to obtain error as $h^4 f^{(4)}(\xi)/12$.

x. **C** $I = \{[f(0) + 4f(1/2) + f(1)]/6\} = 7/20$.

y. **B** Upper case of 'g' is taken as 'case G'.

PART I

Q 2a Perform the operations $R_2 + 2R_1, R_3 - (R_1/2), R_4 - (R_1/2)$.

$$\left[\begin{array}{cccc|c} 2 & 1 & -4 & 1 & 4 \\ 0 & 5 & -3 & 0 & -2 \\ 0 & -3/2 & 3 & -3/2 & 0 \\ 0 & 5/2 & -1 & 3/2 & -3 \end{array}\right]$$. Now, perform the operations $R_3 + (3R_2/10), R_4 - (R_2/2)$.

$$\left[\begin{array}{cccc|c} 2 & 1 & -4 & 1 & 4 \\ 0 & 5 & -3 & 0 & -2 \\ 0 & 0 & 21/10 & -3/2 & -3/5 \\ 0 & 0 & 1/2 & 3/2 & -2 \end{array}\right]$$. Perform $R_4 - (5R_3/21)$.

$$\left[\begin{array}{cccc|c} 2 & 1 & -4 & 1 & 4 \\ 0 & 5 & -3 & 0 & -2 \\ 0 & 0 & 21/10 & -3/2 & -3/5 \\ 0 & 0 & 0 & 39/21 & -13/7 \end{array}\right]$$. $x_4 = -1,\ x_3 = -1,\ x_2 = -1,\ x_1 = 1.$

Q 2b $A = LL^T$. We get $L = \begin{bmatrix} \sqrt{8} & 0 & 0 \\ -3/\sqrt{8} & \sqrt{55/8} & 0 \\ 0 & -3\sqrt{8/55} & \sqrt{368/55} \end{bmatrix}$

$Ax = LL^T x = b$. Set $L^T x = Z$. Solve $LZ = b$, then solve $L^T x = Z$.

$LZ = b$ gives $Z = [-0.35355 \quad 1.02497 \quad 0.64667]^T$

$L^T x = Z$ gives $x = [0.0625 \quad 0.5 \quad 0.25]^T$

Q 3a Set $\varepsilon_k = \xi - x_k$. Substitute in the Newton-Raphson method and expand the terms in Taylor series to obtain

$$\begin{aligned} \varepsilon_{k+1} &= \varepsilon_k - \frac{[\varepsilon_k f'(\xi) + (1/2)\varepsilon_k^2 f''(\xi) + \ldots]}{f'(\xi) + \varepsilon_k f''(\xi) + \ldots} \\ &= \varepsilon_k - [\varepsilon_k + c\varepsilon_k^2 + \ldots][1 + 2c\varepsilon_k + \ldots]^{-1} \\ &= \varepsilon_k - [\varepsilon_k + c\varepsilon_k^2 + \ldots][1 - 2c\varepsilon_k + \ldots] = c\varepsilon_k^2 + \cdots \end{aligned}$$

where $c = f''(\xi)/[2f'(\xi)]$. Hence, $\varepsilon_{k+1} = c\varepsilon_k^2$. The order of the method is $p = 2$.

Q 3b $\varepsilon_{k+1} = c\varepsilon_k^2$, where $c = f''(\xi)/[2f'(\xi)]$. Substituting $k = 0, 1, 2, \ldots$, we get

$\varepsilon_k = c^{2^k-1}\varepsilon_0^{2^k}$ and $|\varepsilon_k| = |c|^{2^k-1}|\varepsilon_0|^{2^k}$.

Here, $|\varepsilon_0| < 1$ and $|c| = \left|\dfrac{f''(\xi)}{2f'(\xi)}\right| \le \dfrac{1}{20}$.

Find k such that $|\varepsilon_k| \le (0.05)^{2^k-1} \le 5\times10^{-7}$. We get $k = 3$.

Q 4a Root lies in $(-2, -1)$. $x_{k+1} = \dfrac{x_{k-1}f(x_k) - x_k f(x_{k-1})}{f(x_k) - f(x_{k-1})}$. $x_0 = -2,\ x_1 = -1$.

We get $x_2 = -1.05263, x_3 = -1.16024, x_4 = -1.14096, x_5 = -1.14282, x_6 = -1.14286$.

Q 4b

```
# include  <stdio.h>
# include  <math.h>
float f(float);
main ( )
{
   float a, b, x, eps, fa, fb, fx;
   int   i, n;
   printf ("Input two approximations to the root \ n");
   scanf ("%f  %f ",&a, &b);
   printf ("input m: number of iterations \ n");
   scanf ("% d", &n);
   printf ("eps: error tolerance \ n");
   scanf ("%E", &eps);
   for (i = 1; i <=  n; i++ )
   {
          fa = f(a);
          fb = f(b);
          x = (a* fb - b*fa)/(fb - fa);
          fx = f(x);
          if  (fabs (fx) <= eps)
                 goto l0;
          a = b;
          b = x;
    }
    printf (" \ n Number of iterations given are not sufficient");
    goto l5;
l0:  printf ("Number of iterations = % d \ n", i);
    printf ("Root  =% 10.7f, f(x) = % e \ n", x, fx);

l5:          return 0;

}
float f(float x)
{
          float fun;
          fun = x*x;
          return (fun);
}
```

Q 5a $x_1^{(k+1)} = \dfrac{1}{4}(2.5 + x_2^{(k)})$; $x_2^{(k+1)} = \dfrac{1}{4}(-2.75 + x_1^{(k)} + x_3^{(k)})$;

$x_3^{(k+1)} = \frac{1}{4}(1.75 + x_2^{(k)} + x_4^{(k)})$; $x_4^{(k+1)} = \frac{1}{4}(-1.25 + x_3^{(k)})$.

$x_1^{(0)} = 0.4, x_2^{(0)} = -0.6, x_3^{(0)} = 0.3, x_4^{(0)} = -0.3$. We get

$x_1^{(1)} = 0.475, x_2^{(1)} = -0.5125, x_3^{(1)} = 0.2125, x_4^{(1)} = -0.2375;$

$x_1^{(2)} = 0.4969, x_2^{(2)} = -0.5156, x_3^{(2)} = 0.25, x_4^{(2)} = -0.2594;$

$x_1^{(3)} = 0.4961, x_2^{(3)} = -0.5008, x_3^{(3)} = 0.2438, x_4^{(3)} = -0.25.$

Q 5b Iteration matrix = $H = -D^{-1}(L+U) = -\begin{bmatrix} 0 & -1/4 & 0 & 0 \\ -1\backslash 4 & 0 & -1\backslash 4 & 0 \\ 0 & -1\backslash 4 & 0 & -1\backslash 4 \\ 0 & 0 & -1\backslash 4 & 0 \end{bmatrix}$

Eigen vaules : $|H - \lambda I| = 0$ gives $256\lambda^4 - 48\lambda^2 + 1 = 0.$

Hence, $\lambda^2 = 0.163627,\ 0.02387$ or $\lambda = \pm 0.4045, \pm 0.1545$.

Spectral radius of $H = \rho(H) = 0.4045.$

Rate of convergence $= -\log_{10}(\rho(H)) = 0.3931.$

Q 6a $f(x, y, z) = x^2 + y^2 - 4.82 = 0, g(x, y, z) = xy + yz + zx - 0.59 = 0,$

$h(x, y, z) = yz^2 + y^2 z + 1.33 = 0.$ Newton's method:

$$\begin{bmatrix} \partial f/\partial x & \partial f/\partial y & \partial f/\partial z \\ \partial g/\partial x & \partial g/\partial y & \partial g/\partial z \\ \partial h/\partial x & \partial h/\partial y & \partial h/\partial z \end{bmatrix}_k \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = -\begin{bmatrix} f \\ g \\ h \end{bmatrix}_k .$$

Next approximation is $x_{k+1} = x_k + \Delta x;\ y_{k+1} = y_k + \Delta y;\ z_{k+1} = z_k + \Delta z.$

$$\begin{bmatrix} 2x & 2y & 0 \\ y+z & z+x & x+y \\ 0 & z^2 + 2yz & y^2 + 2yz \end{bmatrix}_k \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = -\begin{bmatrix} f \\ g \\ h \end{bmatrix}_k$$

Using the given initial approximation, $x_0 = 1,\ y_0 = 2,\ z_0 = -0.5,$ we get

$$\begin{bmatrix} 2 & 4 & 0 \\ 1.5 & 0.5 & 3 \\ 0 & -1.75 & 2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = -\begin{bmatrix} 0.18 \\ -0.09 \\ -0.17 \end{bmatrix}_k$$

The solution of the system is $\Delta x = 0.39,\ \Delta y = -0.24,\ \Delta z = -0.125$.

Hence, $x_1 = 1.39,\ y_1 = 1.76,\ z_1 = -0.625$.

Q 6b

```
# include <stdio.h>
#define  SIZE 100
main ( )
{
        int i, n, f [SIZE];
        void reordering (int n, int f [ ] );
        printf (" \ n Total number of numbers to be entered");
        scanf (" % d", & n);
        printf (" \n");
        for ( i = 0; i < n ; ++i )
        {
                printf ( " i = % d, f = ", i +1);
                scanf (" % d", & f [ i ] );
        }
        reordering  (n, f );
```

```
        printf ( " \ n \ n Numbers in ascending order : \ n \ n" );
        for ( i = 0; i < n; ++i )
                printf ( " i = % d, f = % d \n ", i +1, f[ i ] );

}

void reordering (int n, int f [ ] )
{
        int i , item, temp;
        for (item = 0; item < n - 1; ++item)
                for ( i = item +1; i < n; ++i )
                        if ( f [ i ] < f [ item ] )
                        {
                                temp = f [ item ];
                                f [ item ] = f [ i ];
                                f [ i ] = temp;
                        }
        return;
}
```

PART II

Q 7a Data is not equispaced . Use the divided difference formula.

$$f(x) = f(x_0) + (x-x_0)\, f[x_0, x_1] + (x-x_0)(x-x_1)\, f[x_0, x_1, x_2] + (x-x_0)(x-x_1)(x-x_2)\, f[x_0, x_1, x_2, x_3].$$

Form the divided difference table.

x	f(x)	first d.d	second d.d	third d.d	fourth d.d
0	–2.5				
1	–0.5	2.0			
2	10.5	11.0	4.5		
5	187.5	59.0	12.0	1.50	
7	515.5	164.0	21.0	1.50	0.0
10	1502.5	329.0	33.0	1.50	0.0

$$f(x) = -2.5 + (x-0)2 + (x)(x-1)4.5 + (x)(x-1)(x-2)(1.5) = 1.5x^3 + 0.5x - 2.5$$

$f(8) = 769.5$.

Q 7b $|E| \le \dfrac{h^3}{9\sqrt{3}} M_3$ where $M_3 = \max_{[0,1]} |f'''(x)|$. $h = 0.1,\ M_3 = \max_{[0,1]} (e^{x+1}) = e^2$.

$$|E| \le \frac{(0.1)^3}{9\sqrt{3}} e^2 \approx 0.00047.$$

Q 8a $J = \sum_{i=1}^{N} [\, f(t_i) - (a + b\cos t_i)]^2 = \text{minimum}$

For minimum, we have the necessary conditions

$$\frac{\partial J}{\partial a} = 0 = \sum_{i=1}^{N} [\, f(t_i) - (a + b\cos t_i)], \qquad \frac{\partial J}{\partial b} = 0 = \sum_{i=1}^{N} [\, f(t_i) - (a + b\cos t_i)]\cos t_i.$$

Normal equations are $Na + b\sum \cos t_i = \sum f(t_i)$, $a\sum \cos t_i + b\sum \cos^2 t_i = \sum f(t_i)\cos t_i$.

We get $\sum \cos t_i = 0.2713,\ \sum \cos^2 t_i = 1.8821,\ \sum f(t_i) = 1.4535,\ \sum f(t_i)\cos t_i = 1.4794.$

Normal equations are $5a + 0.2713b = 1.4535$, $0.2713a + 1.8821b = 1.4794$.

Solution is $a = 0.25,\ b = 0.75$. Hence, $f(t) = 0.25 + 0.75\cos t.$

Q 8b

```
# include  <stdio.h>
# include  <math.h>
main ( )
{
        float x[10], y[10], xin, yout, sum;
        int  n, i, j;
        printf ("Input number of points: n \ n");
        scanf ( "%d" , & n);
        printf ( "Input the abscissas \ n");
        for ( i = 1; i <= n; i++)
                scanf ( "% f ", & x[ i ] );
        printf ( "Input the ordinates \ n");
        for ( i = 1; i <= n; i++)
                scanf ( "%f ",  & y[ i ] );
        printf ( "Input the value x for which interpolation is required \ n" );
        scanf ( "% f ",  & xin );
        yout = 0.0 ;
        for (i = 1; i <= n; i++ )
        {
                sum = y[ i ];
                for (j = 1; j <= n; j++ )
                {
                        if  ( i != j )
                                sum = sum * ( xin - x [ j  ] ) / ( x [ i ] - x [ j ] ) ;
                }
                yout = yout  + sum;
        }
        printf ( " At x = % 5.3 f ,  y = % 8.5 f \ n" , xin, yout ) ;
        return 0;
}
```

Q 9a

$$h f'(x_2) = a f(x_2 - 2h) + b f(x_2 - h) + c f(x_2)$$

$$= a\left[f_2 - 2h f_2' + \frac{(2h)^2}{2!} f_2'' - \frac{(2h)^3}{6} f_2''' + ... \right] + b\left[f_2 - h f_2' + \frac{h^2}{2!} f_2'' - \frac{h^3}{6} f_2''' + ... \right] + c f_2$$

$$= (a+b+c) f_2 - h f_2'(2a+b) + \frac{h^2}{2} f_2''(4a+b) - \frac{h^3}{6} f_2'''(8a+b) + ...$$

(i) Comparing, we get $a + b + c = 0$, $2a + b = -1$, $4a + b = 0$.
The solution is $a = 1/2$, $b = -2$, $c = 3/2$.

(ii) Error term = $\frac{h^3}{6}(8a+b) f_2'''(\xi) = \frac{h^3}{3} f_2'''(\xi)$

(iii) Formula is given by $f'(x_2) = \frac{1}{2h}[f(x_0) - 4f(x_1) + 3f(x_2)]$.

Round off error $RE = \frac{1}{2h}[\varepsilon_0 - 4\varepsilon_1 + 3\varepsilon_2]$

$$\therefore |RE| \le \frac{1}{2h}[|\varepsilon_0| + 4|\varepsilon_1| + 3|\varepsilon_2|] \le \frac{4\varepsilon}{h}.$$

Q 9b

$$h = 0.4 : f''(0.6) = \frac{1}{0.16}[f(0.2) - 2f(0.6) + f(1.0)] = 4.64.$$

$$h = 0.2 : f''(0.6) = \frac{1}{0.04}[f(0.4) - 2f(0.6) + f(0.8)] = 4.4.$$

Error : $O(h^2)$. Richardson's estimate $= \frac{1}{3}\left[4f''(h/2) - f''(h)\right]$

$= \frac{1}{3}\left[4f''(h=0.2) - f''(h=0.4)\right] = 4.32$.

Q 10a Make the formula exact for $f(x) = x^i$, $i = 0, 1, 2$.

$f(x) = 1: 2 = a + b$; $f(x) = x: 0 = -a + bc$; $f(x) = x^2: (2/3) = a + bc^2$.

Solution is $a = 1/2,\ b = 3/2,\ c = 1/3$.

Error term: Set $f(x) = x^3 : C = \int_{-1}^{1} x^3 dx - [-a + bc^3] = \frac{4}{9}$.

Error $= \frac{C}{3!} f'''(\xi) = \frac{2}{27} f'''(\xi)$.

Q 10b

```
# include  <stdio.h>
# include  <math.h>
float f(float,float);
main ( )
{
        float x0,y0,h,xf,x,y;
        int  i, iter ;
        FILE *fp ;
        fp = fopen ( "result", "w" );
        printf ( "Input initial  point  initial value , \ n");
        printf ( "Step size h and final value xf \ n");
        scanf ( "%f  %f  %f  %f ", &x0, &y0, &h, &xf );
        fprintf ( fp , "x0  = %f ",  x0 );
        fprintf ( fp , "y0  = %f,  h = %f ", y0, h );
        fprintf ( fp, "Final value = % f \ n", xf );
        iter = (xf-x0)/h+1;
        for ( i = 1; i <= iter ; i++ );
        {
                y=y0+h*f(x0, y0);
                x = x0 + h ;
                if(x<xf)
                {
                        x0  = x ;
                        y0 = y ;
                }
        }
        fprintf ( fp , " At x = % 6.4 f, y =% 12.6 e \ n", x, y );
        printf ( " \ n See FILE `result' for results \ n  \ n" );
        fclose ( fp ) ;
        return 0;
}
float f (float  x, float y)
{
        float fun ;
        fun = x * x + y * y ;
        return ( fun ) ;
}
```

Q 11a (i) Gauss-Legendre 2 –point formula : $\int_{-1}^{1} f(x)dx = f\left(\frac{-1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right)$

$f(x) = (1 - x^2)\cos x$; $I = 2\left(1 - \frac{1}{3}\right)\cos\left(\frac{1}{\sqrt{3}}\right) = 1.1172$.

(ii) Gauss-Chebyshev 2 –point formula :

$$\int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}}\,dx = \frac{\pi}{2}\left[f\left(-\frac{1}{\sqrt{2}}\right) + f\left(\frac{1}{\sqrt{2}}\right)\right]$$

$f(x) = (1-x^2)^{3/2}\cos x;$

$$I = \frac{\pi}{2}(2)\left(1-\frac{1}{2}\right)^{3/2}\cos\left(\frac{1}{\sqrt{2}}\right) = \frac{\pi}{2\sqrt{2}}\cos\left(\frac{1}{\sqrt{2}}\right) = 0.8444.$$

Q 11b (i) Euler method : $u_{n+1} = u_n + h\,f(t_n, u_n) = u_n + h(u_n^2 + t_n^2)$

$t_0 = 0,\ u_0 = 1,\ h = 0.2:$ $\qquad u(0.2) = u(0) + 0.2\,f(0,\ 1) = 1 + 0.2 = 1.2.$

(ii) Taylor series method of order four :

$$u_{n+1} = u_n + h\,u_n' + \frac{h^2}{2}u_n'' + \frac{h^3}{6}u_n''' + \frac{h^4}{24}u_n^{iv}$$

$u_0 = 1,\ u_0' = 1,\ u'' = 2uu' + 2t,\ u_0'' = 2,\ u''' = 2(u\,u'' + u'^2) + 2,$

$u_0''' = 8,\ u^{iv} = 2(u\,u''' + 3u'u'');\ u_0^{iv} = 28.$

$$u(0.2) \approx u_1 = u_0 + hu_0' + \frac{h^2}{2}u_0'' + \frac{h^3}{6}u_0''' + \frac{h^4}{24}u_0^{iv}$$

$$= 1 + 0.2 + 0.02(2) + \frac{0.004}{3}(8) + \frac{0.0004}{6}(28) = 1.2525.$$