CHAPTER

10 Numerical Methods for Unconstrained Optimum Design

Section 10.3 Descent Direction and Convergence of Algorithms

10.1

Answer True or False.

1. All optimum design algorithms require a starting point to initiate the iterative process. *True*
2. A vector of design changes must be computed at each iteration of the iterative process. *True*
3. The design change calculation can be divided into step size determination and direction finding subproblems. *True*
4. The search direction requires evaluation of the gradient of the cost function. *True*
5. Step size along the search direction is always negative. *False*
6. Step size along the search direction can be zero. *False*
7. In unconstrained optimization, the cost function can increase for an arbitrary small step along the descent direction. *False*
8. A descent direction always exists if the current point is not a local minimum. *True*
9. In unconstrained optimization, a direction of descent can be found at a point where the gradient of the cost function is zero. *False*
10. The descent direction makes an angle of 0–90° with the gradient of the cost function. *False*

10.2

Determine whether the given direction at the point is that of descent for the following function (show all of the calculations).

$f(\mathbf{x}) = 3x_1^2 + 2x_1 + 2x_2^2 + 7$; $\mathbf{d}$ = [−1, 1] at $\mathbf{x}$ = [2, 1]

Solution:

$c_1(2,1) = \partial f/\partial x_1 = 6x_1 + 2 = 14; c_2(2,1) = \partial f/\partial x_2 = 4x_2 = 4$

$\mathbf{c} \bullet \mathbf{d} = [14, 4] \bullet [-1,1] = 14(-1) + 4(1) = -10 < 0;$ Thus, $\mathbf{d}$ is descent direction.

10.3

Determine whether the given direction at the point is that of descent for the following function (show all of the calculations).

$f(\mathbf{x}) = x_1^2 + x_2^2 - 2x_1 - 2x_2 + 4$; **d** = [2, 1] at **x** = [1, 1]

Solution:

$c_1(1,1) = \partial f / \partial x_1 = 2x_1 - 2 = 0; c_2(1,1) = \partial f / \partial x_2 = 2x_2 - 2 = 0$

$\mathbf{c} \bullet \mathbf{d} = [0,0] \bullet [2,1] = 0(2) + 0(1) = 0;$ Thus, **d** is not a descent direction.

10.4

Determine whether the given direction at the point is that of descent for the following function (show all of the calculations).

$f(\mathbf{x}) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3$; **d** = [−3, 10, −12] at **x** = [1, 2, 3]

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [2\,x_1 + 2x_2, 4x_2 + 2x_1 + 2x_3, 4x_3 + 2x_2] = [6,16,16]$

$\mathbf{c} \bullet \mathbf{d} = [6,16,16] \bullet [-3,10,-12] = 6(-3) + 16(10) + 16(-12) = -50 < 0;$ Thus, **d** is a descent direction.

10.5

Determine whether the given direction at the point is that of descent for the following function (show all of the calculations).

$f(\mathbf{x}) = 0.1x_1^2 + x_2^2 - 10$; **d** = [1, 2] at **x** = [4, 1]

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [0.2\,x_1, 2x_2] = [0.8, 2]$

$\mathbf{c} \bullet \mathbf{d} = [0.8, 2] \bullet [1, 2] = 0.8(1) + 2(2) = 4.8 > 0;$ Thus, **d** is not a descent direction.

10.6

Determine whether the given direction at the point is that of descent for the following function (show all of the calculations).

$f(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2$; **d** = [2, 3] at **x** = [4, 3]

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [2\,x_1 - 4, 2x_2 - 2] = [4, 4]$

$\mathbf{c} \bullet \mathbf{d} = [4, 4] \bullet [2, 3] = 4(2) + 4(3) = 20 > 0;$ Thus, **d** is not a descent direction.

10.7———————————————————————————

—

Determine whether the given direction at the point is that of descent for the following functions (show all of the calculations).

$f(\mathbf{x}) = 10(x_2 - x_1^2)^2 + (1 - x_1)^2$; **d** = [162, -40] at **x** = [2, 2]

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [20(x_2 - x_1^2)(-2x_1) - 2(1 - x_1), 20(x_2 - x_1^2)] = [162, -40]$

$\mathbf{c} \bullet \mathbf{d} = [162, -40] \bullet [162, -40] = 162(162) - 40(-40) = 27844 > 0;$ Thus, **d** is not a descent direction.

10.8———————————————————————————

Determine whether the given direction at the point is that of descent for the following functions (show all of the calculations).

$f(\mathbf{x}) = (x_1 - 2)^2 + x_2^2;\ \ \mathbf{d} = [-2, 2]$ at $\mathbf{x} = [1, 1]$

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [2(x_1 - 2), 2x_2] = [-2, 2]$

$\mathbf{c} \bullet \mathbf{d} = [-2, 2] \bullet [-2, 2] = -2(-2) + 2(2) = 8 > 0;$ Thus, **d** is not a descent direction.

10.9———————————————————————————

Determine whether the given direction at the point is that of descent for the following functions (show all of the calculations).

$f(\mathbf{x}) = 0.5x_1^2 + x_2^2 - x_1x_2 - 7x_1 - 7x_2;\ \mathbf{d} = [7, 6]$ at $\mathbf{x} = [1, 1]$

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [x_1 - x_2 - 7, 2x_2 - x_1 - 7] = [-7, -6]$

$\mathbf{c} \bullet \mathbf{d} = [-7, -6] \bullet [7, 6] = -7(7) - 6(6) = -85 < 0;$ Thus, **d** is a descent direction.

10.10———————————————————————————

Determine whether the given direction at the point is that of descent for the following functions (show all of the calculations).

$f(\mathbf{x}) = (x_1 + x_2)^2 + (x_2 + x_3)^2;\ \ \mathbf{d} = [4, 8, 4]$ at $\mathbf{x} = [1, 1, 1]$

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [2(x_1 + x_2), 2(x_1 + x_2) + 2(x_2 + x_3), 2(x_2 + x_3)] = [4, 8, 4]$

$\mathbf{c} \bullet \mathbf{d} = [4, 8, 4] \bullet [4, 8, 4] = 4(4) + 8(8) + 4(4) = 96 > 0;$ Thus, **d** is not a descent direction.

10.11

Determine whether the given direction at the point is that of descent for the following functions (show all of the calculations).

$$f(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2;\ \ \mathbf{d} = [2, 4, -2] \text{ at } \mathbf{x} = [1, 2, -1]$$

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [2x_1, 2x_2, 2x_3] = [2, 4, -2]$

$\mathbf{c} \bullet \mathbf{d} = [2, 4, -2] \bullet [2, 4, -2] = 2(2) + 4(4) - 2(-2) = 24 > 0;$ Thus, **d** is not a descent direction.

10.12

Determine whether the given direction at the point is that of descent for the following functions (show all of the calculations).

$$f(\mathbf{x}) = (x_1 + 3x_2 + x_3)^2 + 4(x_1 - x_2)^2;\ \ \mathbf{d} = [-2, -6, -2] \text{ at } \mathbf{x} = [-1, -1, -1]$$

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [2(x_1 + 3x_2 + x_3) + 8(x_1 - x_2), 6(x_1 + 3x_2 + x_3) - 8(x_1 - x_2), 2(x_1 + 3x_2 + x_3)] = [-10, -30, -10]$

$\mathbf{c} \bullet \mathbf{d} = [-10, -30, -10] \bullet [-2, -6, -2] = -10(-2) - 30(-6) - 10(-2) = 220 > 0$

Thus, **d** is not a descent direction.

10.13

Determine whether the given direction at the point is that of descent for the following functions (show all of the calculations).

$$f(\mathbf{x}) = 9 - 8x_1 - 6x_2 - 4x_3 - 2x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_2x_3;\ \ \mathbf{d} = [-2, 2, 0] \text{ at } \mathbf{x} = [1,1,1]$$

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [-8 - 4x_1 + 2x_2, -6 + 4x_2 + 2x_1 + 2x_3, -4 + 2x_3 + 2x_2] = [-1, 2, 0]$

$\mathbf{c} \bullet \mathbf{d} = [-1, 2, 0] \bullet [-2, 2, 0] = -1(-2) + 2(2) + 0(0) = 6 > 0;$ Thus, **d** is not a descent direction.

10.14

Determine whether the given direction at the point is that of descent for the following functions (show all of the calculations).

$$f(\mathbf{x}) = (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2 + (x_4 - 4)^2;\ \ \mathbf{d} = [2, -2, 2, -2] \text{ at } \mathbf{x} = [2, 1, 4, 3]$$

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [2(x_1 - 1), 2(x_2 - 2), 2(x_3 - 3), 2(x_4 - 4)] = [2, -2, 2, -2]$

$\mathbf{c} \bullet \mathbf{d} = [2, -2, 2, -2] \bullet [2, -2, 2, -2] = 2(2) - 2(-2) + 2(2) - 2(-2) = 16 > 0;$

Thus, **d** is not a descent direction.

Section 10.5 Numerical Methods to Compute Step Size

10.15

Answer True or False.

1. Step size determination is always a one-dimensional problem. *True*
2. In unconstrained optimization, the slope of the cost function along the descent direction at zero step size is always positive. *False*
3. The optimum step lies outside the interval of uncertainty. *False*
4. After initial bracketing, the golden section search requires two function evaluations to reduce the interval of uncertainty. *False*

10.16

Find the minimum of the function $f(\alpha)=7\alpha^2-20\alpha+22$ using the equal-interval search method within an accuracy of 0.001. Use $\delta=0.05$.

Solution:

The problem is solved using the example program shown below, which was written in C++. This code can be copied and pasted into a compiler such as Code::Blocks. Or, this code can be modified into your language of choice.

```
#include <iostream>
#include <math.h>

using namespace std;

void equalInterval(float *a, float d, float e, float *f, int *n);
float funct(float a, int *nC);

int main()
{
    float delta = 0.05;         //modify this value for the required delta
    float epsilon = 0.001;      //modify this value for the required accuracy
    int nCount = 0;             //tracks number of function evaluations
    float f = 0;                //holds the value of the function at the minimum
    float alpha = 0;            //holds the value of alpha at the minimum

    float *alphaptr = &alpha;   //pointer to alpha
    float *fptr = &f;           //pointer to f
    int *nptr = &nCount;        //point to nCount

    //to perform equal interval line search call function equalInterval
    equalInterval(alphaptr, delta, epsilon, fptr, nptr);

    cout <<"Minimum = "<<alpha<<"\n";              //outputs minimum value of alpha
    cout <<"Minimum function value = "<<f<<"\n";   //outputs minimum function value
    cout <<"No. of function evaluations = "<<nCount<<"\n";     //outputs the number
    of function evaluations

    return 0;
}

/* this function implements equal interval search
```

```
    a = optimal value of alpha on return
    d = initial step length
    e = convergence parameter
    f = optimum value of the function on return
    n = number of function evaluations on return
**/
void equalInterval(float *a, float d, float e, float *f, int *n)
{
    int q = 2;      //delta multiplier
    float al = 0;   //lower bound of alpha
    float fa = 0;   //holds function value of mid-point alpha
    float aa = d;   //holds value of mid-point alpha, initially set to delta
    float fu = 0;   //holds function value of upper alpha bound
    float au = q*d; //holds value of upper bound of alpha, initially set to 2*delta

    fa = funct(aa, n);

    //establish initial interval of uncertainty
    do
    {
        fu = funct(au, n);
        if( fa >= fu )
        {
            q = q+1;
            aa = au;
            fa = fu;
            au = q*d;
        }
    }while(fa >= fu);

    al = (q-2)*d;

    //refine the interval of uncertainty further
    while((au-al) > e)
    {
        d = d*0.1;              //refine delta value
        q = 2;
        aa = al+d;
        au = al+(q*d);
        fa = funct(aa, n);

        do
        {
            fu = funct(au, n);
            if(fa >= fu)
            {
                q = q+1;
                aa = au;
                fa = fu;
                au = al+(q*d);
            }
        }while(fa >= fu);

        al = al+((q-2)*d);
    }

    *a = (au+al)/(float)2;      //send minimum alpha value to pointer address
    *f = funct(*a, n);          //send minimum function value to pointer address
}

/* calculates the function value
    a = value of alpha, input
    fVal = function value on return
    nC = number of calls for function evaluation
**/
float funct(float a, int *nC)
{
    float fVal = 0;
    *nC = *nC + 1;              //send function evaluation count to pointer address
    fVal = (7*(a*a))-(20*a)+22; //enter function of one variable here
    return fVal;
}
```

Output: Minimum = 1.4285;
Minimum function value = 7.71429;
No. of function evaluations = 46.

10.17

For the function f(α)=7α^2−20α+22, use the golden section method to find the minimum with an accuracy of 0.005 (final interval of uncertainty should be less than 0.005). Use δ=0.05.

Solution:

The problem is solved using the example program shown below, which was written in C++. This code can be copied and pasted into a compiler such as Code::Blocks. Or, this code can be modified into your language of choice.

```
#include <iostream>
#include <math.h>

using namespace std;

void goldenInterval(float *a, float d, float e, float *f, int *n);
float funct(float a, int *nC);

int main()
{
    float delta = 0.05;         // * modify this value for the required delta
    float epsilon = 0.005;      // * modify this value for the required accuracy
    int nCount = 0;             //tracks number of function evaluations
    float f = 0;                //holds the value of the function at the minimum
    float alpha = 0;            //holds the value of alpha at the minimum

    float *alphaptr = &alpha;   //pointer to alpha
    float *fptr = &f;           //pointer to f
    int *nptr = &nCount;        //point to nCount

    //to perform equal interval line search call function equalInterval
    goldenInterval(alphaptr, delta, epsilon, fptr, nptr);

    cout <<"Minimum = "<<alpha<<"\n";                //outputs minimum value of alpha
    cout <<"Minimum function value = "<<f<<"\n";   //outputs minimum function value
    cout <<"No. of function evaluations = "<<nCount<<"\n";     //outputs the number
    of function evaluations

    return 0;
}

/* this function implements golden interval search
    a = optimal value of alpha on return
    d = initial step length
    e = convergence parameter
    f = optimum value of the function on return
    n = number of function evaluations on return
**/
void goldenInterval(float *a, float d, float e, float *f, int *n)
{
    int q = 1;            //delta multiplier
    float al = 0;         //lower bound of alpha
    float aa = d;         //holds value of mid-point alpha a, initially set to delta
    float ab = 0;         //holds value of mid-point alpha b
    float au = d+(pow((float)1.618, q)*d);  //holds value of upper bound of alpha
    float fa = 0;                       //holds function value of mid-point alpha
    float fb = 0;                       //holds function value at mid-point alpha b
    float fu = 0;                       //holds function value of upper alpha bound

    fa = funct(aa, n)             //call function funct() to return the value at a
    given value of alpha
    fu = funct(au, n);

    //establish initial interval of uncertainty
    while(fa >= fu)
    {
        q = q+1;
        al = aa;
        aa = au;
        au = au+(pow((float)1.618, q)*d);
        fa = funct(aa, n);
        fu = funct(au, n);
    }

    ab = al+((float)0.618*(au-al));
    fb = funct(ab, n);

    //refine the interval of uncertainty further
    while((au-al) >= e)
    {
        int caseInPoint = 0;

        if(fa < fb)
            caseInPoint = 1;
        else if(fa > fb)
            caseInPoint = 2;
        else if(fa == fb)
            caseInPoint = 3;

        switch(caseInPoint)
        {
            case 1:
                au = ab;
                ab = aa;
                fb = fa;
                aa = al+((float)0.382*(au-al));
                fa = funct(aa, n);
                break;
            case 2:
                al = aa;
                aa = ab;
                fa = fb;
                ab = al+((float)0.618*(au-al));
                fb = funct(ab, n);
                break;
            case 3:
                al = aa;
                au = ab;
                aa = al+((float)0.382*(au-al));
                ab = al+((float)0.618*(au-al));
                fa = funct(aa, n);
                fb = funct(ab, n);
```

```
                break;
            default:
                break;
        }
    }

    *a = (au+al)/(float)2;      //send minimum alpha value to pointer address
    *f = funct(*a, n);          //send minimum function value to pointer address
}

/* calculates the function value
    a = value of alpha, input
    fVal = function value on return
    nC = number of calls for function evaluation
**/
float funct(float a, int *nC)
{
    float fVal = 0;
    *nC = *nC + 1;              //send function evaluation count to pointer address
    fVal = (7*(a*a))-(20*a)+22; //enter function of one variable here
    return fVal;
}
```

Output:
```
Minimum = 1.42743;
Minimum function value = 7.71429;
No. of function evaluations = 26.
```

10.18

Write a computer program to implement the alternate equal-interval search process shown in Figure 10.7 for any given function f(α). For the function $f(\alpha)=2-4\alpha+e^{\alpha}$, use your program to find the minimum within an accuracy of 0.001. Use $\delta=0.50$.

Solution:

The problem is solved using the example program shown below, which was written in C++. This code can be copied and pasted into a compiler such as Code::Blocks. Or, this code can be modified into your language of choice.

```
#include <iostream>
#include <math.h>

using namespace std;

void altEqualInterval(float *a, float d, float e, float *f, int *n);
float funct(float a, int *nC);

int main()
{
    float delta = 0.5;          // * modify this value for the required delta
    float epsilon = 0.001;       // * modify this value for the required accuracy
    int nCount = 0;              //tracks number of function evaluations
    float f = 0;                 //holds the value of the function at the minimum
    float alpha = 0;             //holds the value of alpha at the minimum

    float *alphaptr = &alpha;   //pointer to alpha
    float *fptr = &f;            //pointer to f
    int *nptr = &nCount;         //point to nCount

    //to perform equal interval line search call function equalInterval
    altEqualInterval(alphaptr, delta, epsilon, fptr, nptr);

    cout <<"Minimum = "<<alpha<<"\n";               //outputs minimum value of alpha
    cout <<"Minimum function value = "<<f<<"\n";   //outputs minimum function value
    cout <<"No. of function evaluations = "<<nCount<<"\n";  //outputs the number of
                                                            //function evaluations

    return 0;
}

/* this function implements the alternate equal interval search
    a = optimal value of alpha on return
    d = initial step length
    e = convergence parameter
    f = optimum value of the function on return
    n = number of function evaluations on return
**/
void altEqualInterval(float *a, float d, float e, float *f, int *n)
{
    int q = 2;       //delta multiplier
    float al = 0;    //lower bound of alpha
    float aa = d;    //holds value of mid-point alpha a, initially set to delta
    float ab = 0;   //holds value of mid-point alpha b
    float au = q*d; //holds value of upper bound of alpha, initially set to 2*delta
    float fa = 0;   //holds function value of mid-point alpha
    float fb = 0;   //holds function value at mid-point alpha b
    float fu = 0;   //holds function value of upper alpha bound

    fa = funct(aa, n);  //call function funct() to return the value at a given
                        //value of alpha
    fu = funct(au, n);

    //establish initial interval of uncertainty
    while(fa >= fu)
    {
        q = q+1;
        al = aa;
        aa = au;
        au = q*d;
        fa = funct(aa, n);
        fu = funct(au, n);
    }

    aa = al+((au-al)/(float)3);
    ab = al+(2*((au-al)/(float)3));

    //refine the interval of uncertainty further
    while((au-al) > e)
    {
        fa = funct(aa, n);
        fb = funct(ab, n);

        int caseInPoint = 0;

        if(fa < fb)
            caseInPoint = 1;
        else if(fa > fb)
            caseInPoint = 2;
        else if(fa == fb)
            caseInPoint = 3;

        switch(caseInPoint)
        {
            case 1:
                au = ab;
                aa = al+((au-al)/(float)3);
                ab = al+(2*((au-al)/(float)3));
                break;
            case 2:
                al = aa;
                aa = al+((au-al)/(float)3);
                ab = al+(2*((au-al)/(float)3));
                break;
            case 3:
                al = aa;
                au = ab;
                aa = al+((au-al)/(float)3);
                ab = al+(2*((au-al)/(float)3));
            default:
                break;
        }
    }

    *a = (au+al)/(float)2;      //send minimum alpha value to pointer address
    *f = funct(*a, n);          //send minimum function value to pointer address
}

/* calculates the function value
    a = value of alpha, input
    fVal = function value on return
    nC = number of calls for function evaluation
**/
float funct(float a, int *nC)
{
    float fVal = 0;
    *nC = *nC + 1;                 //send function evaluation count to pointer
address
    fVal = 2-(4*a)+((float)exp( a ));    //enter function of one variable here
    return fVal;
}
```

```
Output:     Minimum = 1.38629;
            Minimum function value = 0.454823;
            No. of function evaluations = 43.
```

10.19

Consider the function $f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3$. *Verify whether the vector* $\mathbf{d} = [-12, -40, -48]$ *at the point* $\mathbf{x} = [2, 4, 10]$ *is a descent direction for f. What is the slope of the function at the given point? Find an optimum step size along* **d** *by any numerical method.*

Solution:

$\mathbf{c} = \nabla f(\mathbf{x}) = [2x_1 + 2x_2, 4x_2 + 2x_1 + 2x_3, 4x_3 + 2x_2] = [12, 40, 48]$

$\mathbf{c} \bullet \mathbf{d} = -4048 < 0;$ Thus, **d** is a descent direction.

The slope of the function is **c•d** = − 4048.

$f(\alpha) = f(\mathbf{x} + \alpha\mathbf{d}) = (2 - 12\alpha)^2 + 2(10 - 48\alpha)^2 + 2(4 - 40\alpha)(16 - 100\alpha)$

$f'(\alpha) = -4048 + 25504\alpha = 0; \alpha^* = 0.15872$

10.20

Consider the function $f(\mathbf{x}) = x_1^2 + x_2^2 - 2x_1 - 2x_2 + 4$. *At the point* $\mathbf{x} = [1,1]$, *let a search direction be defined as* $\mathbf{d} = [1, 2]$. *Express f as a function of one variable at the given point along* **d**. *Find an optimum step size along* **d** *analytically.*

Solution:

$[x_1, x_2] = [1,1] + \alpha[1, 2] = [1 + \alpha, 1 + 2\alpha];$

$f(\alpha) = (1 + \alpha)^2 + (1 + 2\alpha)^2 - 2(1 + \alpha) - 2(1 + 2\alpha) + 4$

$f'(\alpha) = 0$ gives $\alpha = 0$. The optimum step size along **d** is zero.

10.21

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$f(\mathbf{x}) = 0.1x_1^2 + x_2^2 - 10;$ $\mathbf{d} = [-1, -2]$ at $\mathbf{x} = [5,1]$

Solution:

$[x_1, x_2] = [5,1] + \alpha[-1, -2] = [5 - \alpha, 1 - 2\alpha];$ $f(\mathbf{x}) = 0.1x_1^2 + x_2^2 - 10$

$f(\alpha) = 0.1(5 - \alpha)^2 + (1 - 2\alpha)^2 - 10;$ $f(\alpha) = 4.1\alpha^2 - 5\alpha - 6.5$

10.22

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$f(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2;$ $\mathbf{d} = [-4, -6]$ *at* $\mathbf{x} = [4, 4]$

Solution:

$[x_1, x_2] = [4, 4] + \alpha[-4, -6] = [4 - 4\alpha, 4 - 6\alpha];$ $f(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2$

$f(\alpha) = (4 - 4\alpha - 2)^2 + (4 - 6\alpha - 1)^2;$ $f(\alpha) = 52\alpha^2 - 52\alpha + 13$

10.23

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$$f(\mathbf{x}) = 10(x_2 - x_1^2)^2 + (1 - x_1)^2;\ \mathbf{d} = [-162, 40]\ \ at\ \ \mathbf{x} = [2, 2]$$

Solution:

$[x_1, x_2] = [2, 2] + \alpha[-162, 40] = [2 - 162\alpha, 2 + 40\alpha];\ f(\mathbf{x}) = 10(x_2 - x_1^2)^2 + (1 - x_1)^2$

$f(\alpha) = 10(2 + 40\alpha - (2 - 162\alpha)^2)^2 + (-1 + 162\alpha)^2;$

$f(\alpha) = 6887475360\alpha^4 - 361117440\alpha^3 + 5809444\alpha^2 - 27844\alpha + 41$

10.24

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$$f(\mathbf{x}) = (x_1 - 2)^2 + x_2^2;\ \mathbf{d} = [2, -2]\ \ at\ \ \mathbf{x} = [1,1]$$

Solution:

$[\mathrm{x}_1, x_2] = [1,1] + \alpha[2, -2] = [1 + 2\alpha, 1 - 2\alpha;\ f(\mathbf{x}) = (x_1 - 2)^2 + x_2^2$

$f(\alpha) = (1 + 2\alpha - 2)^2 + (1 - 2\alpha)^2;\ f(\alpha) = 8\alpha^2 - 8\alpha + 2$

10.25

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$$f(\mathbf{x}) = 0.5x_1^2 + x_2^2 - x_1x_2 - 7x_1 - 7x_2;\ \mathbf{d} = [7, 6]\ \text{at}\ \mathbf{x} = [1,1]$$

Solution:

$[x_1, x_2] = [1,1] + \alpha[7, 6] = [1 + 7\alpha, 1 + 6\alpha];\ f(\mathbf{x}) = 0.5x_1^2 + x_2^2 - x_1x_2 - 7x_1 - 7x_2$

$f(\alpha) = 0.5(1 + 7\alpha)^2 + (1 + 6\alpha)^2 - (1 + 7\alpha)(1 + 6\alpha) - 7(1 + 7\alpha) - 7(1 + 6\alpha)$

$f(\alpha) = 18.5\alpha^2 - 85\alpha - 13.5$

10.26

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$$f(\mathbf{x}) = (x_1 + x_2)^2 + (x_2 + x_3)^2;\ \mathbf{d} = [-4, -8, -4]\ \ at\ \ \mathbf{x} = [1,1,1]$$

Solution:

$[x_1, x_2, x_3] = [1,1,1] + \alpha[-4, -8, -4] = [1 - 4\alpha, 1 - 8\alpha, 1 - 4\alpha];\ f(\mathbf{x}) = (x_1 + x_2)^2 + (x_2 + x_3)^2$

$f(\alpha) = (1 - 4\alpha + 1 - 8\alpha)^2 + (1 - 8\alpha + 1 - 4\alpha)^2;\ f(\alpha) = 288\alpha^2 - 96\alpha + 8$

10.27

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$$f(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2;\ \mathbf{d} = [-2, -4, 2]\ \ at\ \ \mathbf{x} = [1, 2, -1]$$

Solution:

$[x_1, x_2, x_3] = [1, 2, -1] + \alpha[-2, -4, 2] = [1 - 2\alpha, 2 - 4\alpha, -1 + 2\alpha];\ f(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2$

$f(\alpha) = (1 - 2\alpha)^2 + (2 - 4\alpha)^2 + (-1 + 2\alpha)^2;\ f(\alpha) = 24\alpha^2 - 24\alpha + 6$

10.28

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$$f(\mathbf{x}) = (x_1 + 3x_2 + x_3)^2 + 4(x_1 - x_2)^2;\ \mathbf{d} = [1, 3, 1]\ \ at\ \ \mathbf{x} = [-1, -1, -1]$$

Solution:

$[x_1, x_2, x_3] = [-1, -1, -1] + \alpha[1, 3, 1] = [-1 + \alpha, -1 + 3\alpha, -1 + \alpha];\ f(\mathbf{x}) = (x_1 + 3x_2 + x_3)^2 + 4(x_1 - x_2)^2$

$f(\alpha) = (-1 + \alpha - 3 + 9\alpha - 1 + \alpha)^2 + 4(-1 + \alpha + 1 - 3\alpha)^2;\ f(\alpha) = 137\alpha^2 - 110\alpha + 25$

10.29

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$$f(\mathbf{x}) = 9 - 8x_1 - 6x_2 - 4x_3 + 2x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_2x_3;\ \mathbf{d} = [2, -2, 0]\ \text{at}\ \mathbf{x} = [1,1,1]$$

Solution:

$[x_1, x_2, x_3] = [1,1,1] + \alpha[2, -2, 0] = [1 + 2\alpha, 1 - 2\alpha, 1]$

$f(\mathbf{x}) = 9 - 8x_1 - 6x_2 - 4x_3 + 2x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_2x_3$

$f(\alpha) = 9 - 8(1 + 2\alpha) - 6(1 - 2\alpha) - 4 + 2(1 + 2\alpha)^2 + 2(1 - 2\alpha)^2 + 1 + 2(1 + 2\alpha)(1 - 2\alpha) + 2(1 - 2\alpha)$

$f(\alpha) = 8\alpha^2 - 8\alpha$

10.30

For the following function, direction of change at a point is given. Derive the function of one variable (line search function) that can be used to determine optimum step size (show all calculations).

$$f(\mathbf{x}) = (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2 + (x_4 - 4)^2;\ \mathbf{d} = [-2, 2, -2, 2]\ \text{at}\ \mathbf{x} = [2, 1, 4, 3]$$

Solution:

$[x_1, x_2, x_3, x_4] = [2, 1, 4, 3] + \alpha[-2, 2, -2, 2] = [2 - 2\alpha, 1 + 2\alpha, 4 - 2\alpha, 3 + 2\alpha];$

$f(\mathbf{x}) = (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2 + (x_4 - 4)^2$

$f(\alpha) = (1 - 2\alpha)^2 + (-1 + 2\alpha)^2 + (1 - 2\alpha)^2 + (-1 + 2\alpha)^2;\ f(\alpha) = 16\alpha^2 - 16\alpha + 4$

Problems 10.31 to 10.40 are solved using a modified form of the program in Section B.2 of Appendix B.

10.31 ———

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = 0.1x_1^2 + x_2^2 - 10;\ \mathbf{d} = [-1, -2]\ \text{at}\ \mathbf{x} = [5, 1]$$

Solution:

$f(\alpha) = 4.1\alpha^2 - 5\alpha - 6.5$ (from the solution to Problem 10.21)

Solved using the following modified program in Section B.2, Appendix B. The code was modified by commenting out the “refining” portion of the code, as shown in the example below, and adding two values to save the upper and lower bounds of the initial alpha value. The same code was used for problems 10.32 – 10.40 (changing the equation in the function funct()), for which example code will not be repeated.

```
#include <iostream>
#include <math.h>

using namespace std;

void equalInterval(float *a, float *aui, float *ali, float d, float e, float *f,
    int *n);
float funct(float a, int *nC);

int main()
{
    float delta = 0.05;         //modify this value for the required delta
    float epsilon = 0.001;      //modify this value for the required accuracy
    int nCount = 0;             //tracks number of function evaluations
    float f = 0;                //holds the value of the function at the minimum
    float alpha = 0;            //holds the value of alpha at the minimum
    float ali = 0;              // holds the initial value of the lower alpha bound
    float aui = 0;              // holds the initial value of the upper alpha bound

    float *alphaptr = &alpha;   //pointer to alpha
    float *aliptr = &ali;       //pointer to ali
    float *auiptr = &aui;       //pointer to aui
    float *fptr = &f;           //pointer to f
    int *nptr = &nCount;        //point to nCount

    //to perform equal interval line search call function equalInterval
    equalInterval(alphaptr, auiptr, aliptr, delta, epsilon, fptr, nptr);

    cout<<"Initial lower alpha bound = "<<ali<<"\n";            //outputs the
    initial lower alpha bound
    cout<<"Initial upper alpha bound = "<<aui<<"\n";            //outputs the
    initial upper alpha bound
    cout<<"Initial interval of uncertainty = "<<aui-ali<<"\n";  //outputs the
    initial interval of uncertainty
    cout<<"Function value at initial minimum = "<<f<<"\n";      //outputs minimum
    function value
    cout <<"No. of function evaluations = "<<nCount<<"\n";      //outputs the
    number of function evaluations

    return 0;
}

/* this function implements equal interval search
    a = optimal value of alpha on return
    d = initial step length
    e = convergence parameter
    f = optimum value of the function on return
    n = number of function evaluations on return
**/
void equalInterval(float *a, float *aui, float *ali, float d, float e, float *f,
    int *n)
{
    int q = 2;               //delta multiplier
    float al = 0;            //lower bound of alpha
    float fa = 0;            //holds function value of mid-point alpha
    float aa = d;            //holds value of mid-point alpha, initially set to
    delta
    float fu = 0;            //holds function value of upper alpha bound
    float au = q*d;          //holds value of upper bound of alpha, initially set to
    2*delta

    fa = funct(aa, n);
    fu = funct(au, n);

    //establish initial interval of uncertainty
    while(fa >= fu)
    {
        q = q+1;
        aa = au;
        fa = fu;
        au = q*d;
        fu = funct(au, n);
    }

    al = (q-2)*d;
    *aui = au;
    *ali = al;

    //refine the interval of uncertainty further
//    while((au-al) > e)
//    {
//        d = d*0.1;               //refine delta value
//        q = 2;
//        aa = al+d;
//        au = al+(q*d);
//        fa = funct(aa, n);
//        fu = funct(au, n);
//
//        while(fa >= fu)
//        {
//            q = q+1;
//            aa = au;
//            fa = fu;
//              au = al+(q*d);
//              fu = funct(au, n);
//          }
//
//          al = al+((q-2)*d);
//      }

    *a = (au+al)/(float)2;      //send minimum alpha value to pointer address
    *f = funct(*a, n);          //send minimum function value to pointer address
}

/* calculates the function value
    a = value of alpha, input
    fVal = function value on return
    nC = number of calls for function evaluation
**/
float funct(float a, int *nC)
{
    float fVal = 0;
    *nC = *nC + 1;              //send function evaluation count to pointer address
    fVal = 4.1*pow(a,2)-5*a-6.5;     //enter function of one variable here
    return fVal;
}
```

Output: Initial lower alpha bound = 0.55;
Initial upper alpha bound = 0.65;
Minimum (α*) after two iterations = 0.1;
Minimum function value (*f**) = -8.024;
No. of function evaluations = 14.

10.32

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2; \ \ \mathbf{d} = [-4, -6] \text{ at } \mathbf{x} = [4, 4]$$

Solution:

$f(\alpha) = 52\alpha^2 - 52\alpha + 13$ (from the solution to Problem 10.22)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 0.45;
          Initial upper alpha bound = 0.55;
          Minimum (α*) after two iterations = 0.1;
          Minimum function value (f*) = 0;
          No. of function evaluations = 12.
```

10.33

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = 10(x_2 - x_1^2)^2 + (1 - x_1)^2; \ \mathbf{d} = [-162, 40] \text{ at } \mathbf{x} = [2, 2]$$

Solution:

$f(\alpha) = 6887475360\alpha^4 - 361117440\alpha^3 + 5809444\alpha^2 - 27844\alpha + 41$ (from the solution to Problem 10.23)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 0;
          Initial upper alpha bound = 0.1;
          Minimum (α*) after two iterations = 0.1;
          Minimum function value (f*) = 11079.5;
          No. of function evaluations = 3.
```

10.34

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = (x_1 - 2)^2 + x_2^2; \ \mathbf{d} = [2, -2] \text{ at } \mathbf{x} = [1, 1]$$

Solution:

$f(\alpha) = 8\alpha^2 - 8\alpha + 2$ (from the solution to Problem 10.24)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 0.45;
          Initial upper alpha bound = 0.55;
          Minimum (α*) after two iterations = 0.1;
          Minimum function value (f*) = 0;
          No. of function evaluations = 12.
```

10.35

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = 0.5x_1^2 + x_2^2 - x_1x_2 - 7x_1 - 7x_2;\ \mathbf{d} = [7, 6] \text{ at } \mathbf{x} = [1,1]$$

Solution:

$f(\alpha) = 18.5\alpha^2 - 85\alpha - 13.5$ (from the solution to Problem 10.25)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 2.25;
Initial upper alpha bound = 2.35;
Minimum (α*) after two iterations = 0.1;
Minimum function value (*f**) = -111.135;
No. of function evaluations = 48.

10.36

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = (x_1 + x_2)^2 + (x_2 + x_3)^2;\ \mathbf{d} = [-4, -8, -4]\ at\ \mathbf{x} = [1,1,1]$$

Solution:

$f(\alpha) = 288\alpha^2 - 96\alpha + 8$ (from the solution to Problem 10.26)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 0.1;
Initial upper alpha bound = 0.2;
Minimum (α*) after two iterations = 0.1;
Minimum function value (*f**) = 0.0799999;
No. of function evaluations = 5.

10.37

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2;\ \mathbf{d} = [-2, -4, 2]\ at\ \mathbf{x} = [1, 2, -1]$$

Solution:

$f(\alpha) = 24\alpha^2 - 24\alpha + 6$ (from the solution to Problem 10.27)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 0.45;
Initial upper alpha bound = 0.55;
Minimum (α*) after two iterations = 0.1;
Minimum function value (*f**) = 0;
No. of function evaluations = 12.

10.38

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = (x_1 + 3x_2 + x_3)^2 + 4(x_1 - x_2)^2; \ \mathbf{d} = [1,3,1] \ \ at \ \ \mathbf{x} = [-1,-1,-1]$$

Solution:

$f(\alpha) = 137\alpha^2 - 110\alpha + 25$ (from the solution to Problem 10.28)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 0.35;
Initial upper alpha bound = 0.45;
Minimum (α*) after two iterations = 0.1;
Minimum function value (*f**) = 2.92;
No. of function evaluations = 10.

10.39

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = 9 - 8x_1 - 6x_2 - 4x_3 + 2x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_2x_3; \ \mathbf{d} = [2,-2,0] \ \text{at} \ \mathbf{x} = [1,1,1]$$

Solution:

$f(\alpha) = 8\alpha^2 - 8\alpha$ (from the solution to Problem 10.29)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 0.45;
Initial upper alpha bound = 0.55;
Minimum (α*) after two iterations = 0.1;
Minimum function value (*f**) = -2;
No. of function evaluations = 12.

10.40

For the following function, calculate the initial interval of uncertainty for the equal-interval search with δ=0.05 at the given point and in the given search direction.

$$f(\mathbf{x}) = (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2 + (x_4 - 4)^2; \ \mathbf{d} = [-2,2,-2,2] \ \text{at} \ \mathbf{x} = [2,1,4,3]$$

Solution:

$f(\alpha) = 16\alpha^2 - 16\alpha + 4$ (from the solution to Problem 10.30)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 0.45;
Initial upper alpha bound = 0.55;
Minimum (α*) after two iterations = 0.1;
Minimum function value (*f**) = 0;
No. of function evaluations = 12.

Problems 10.41 to 10.50 are solved using the program in Section B.2 of Appendix B.

10.41

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = 0.1x_1^2 + x_2^2 - 10;\ \ \mathbf{d} = [-1, -2] \text{ at } \mathbf{x} = [5, 1]$$

Solution:

$f(\alpha) = 4.1\alpha^2 - 5\alpha - 6.5$ (from the solution to Problem 10.21)

Solved using the following modified program in Section B.2, Appendix B. The code was modified to save the initial values of the lower and upper alpha bounds, then complete only two iterations of Phase II, rather than fully refining the function in Phase II. The same program was using for Problems 10.42 – 10.50 (changing the equation in the function funct()), for which example code will not be repeated.

```
#include <iostream>
#include <math.h>

using namespace std;

void goldenInterval(float *a, float *aui, float *ali, float d, float e, float *f,
    int *n);
float funct(float a, int *nC);

int main()
{
    float delta = 0.05;         // * modify this value for the required delta
    float epsilon = 0.001;      // * modify this value for the required accuracy
    int nCount = 0;             //tracks number of function evaluations
    float f = 0;                //holds the value of the function at the minimum
    float alpha = 0;            //holds the value of alpha at the minimum
    float ali = 0;              // holds the initial value of the lower alpha bound
    float aui = 0;              // holds the initial value of the upper alpha bound

    float *alphaptr = &alpha;   //pointer to alpha
    float *aliptr = &ali;       //pointer to ali
    float *auiptr = &aui;       //pointer to aui
    float *fptr = &f;           //pointer to f
    int *nptr = &nCount;        //point to nCount

    //to perform equal interval line search call function equalInterval
    goldenInterval(alphaptr, auiptr, aliptr, delta, epsilon, fptr, nptr);

    cout<<"Initial lower alpha bound = "<<ali<<"\n";
    cout<<"Initial upper alpha bound = "<<aui<<"\n";
    cout<<"Minimum after two iterations = "<<alpha<<"\n";
    //outputs minimum value of alpha
    cout<<"Minimum function value after two iterations = "<<f<<"\n";
    //outputs minimum function value
    cout<<"No. of function evaluations = "<<nCount<<"\n";     //outputs the number
    of function evaluations
    return 0;
}

/* this function implements golden interval search
    a = optimal value of alpha on return
    d = initial step length
    e = convergence parameter
    f = optimum value of the function on return
    n = number of function evaluations on return
**/
void goldenInterval(float *a, float *aui, float *ali, float d, float e, float *f,
    int *n)
{
    int q = 1;           //delta multiplier
    float al = 0;        //lower bound of alpha
    float aa = d;        //holds value of mid-point alpha a, initially set to delta
    float ab = 0;                             //holds value of mid-point alpha b
    float au = d+(pow((float)1.618, q)*d);  //holds value of upper bound of alpha,
    //initially set to 2*delta
    float fa = 0;                       //holds function value of mid-point alpha
    float fb = 0;                       //holds function value at mid-point alpha b
    float fu = 0;                       //holds function value of upper alpha bound

    fa = funct(aa, n);  //call function funct() to return the value at a given
                        //value of alpha
    fu = funct(au, n);

    //establish initial interval of uncertainty
    while(fa >= fu)
    {
        q = q+1;
        al = aa;
        aa = au;
        au = au+(pow((float)1.618, q)*d);
        fa = funct(aa, n);
        fu = funct(au, n);
    }

    *aui = au;
    *ali = al;

    ab = al+((float)0.618*(au-al));
    fb = funct(ab, n);

    //complete two iterations of Phase II
    for(int i = 0; i < 2; i++)
    {
        int caseInPoint = 0;

        if(fa < fb)
            caseInPoint = 1;
        else if(fa > fb)
            caseInPoint = 2;
        else if(fa == fb)
            caseInPoint = 3;

        switch(caseInPoint)
        {
            case 1:
                au = ab;
                ab = aa;
                fb = fa;
                aa = al+((float)0.382*(au-al));
                fa = funct(aa, n);
                break;
            case 2:
                al = aa;
                aa = ab;
                fa = fb;
                ab = al+((float)0.618*(au-al));
                fb = funct(ab, n);
                break;
            case 3:
                al = aa;
                au = ab;
                aa = al+((float)0.382*(au-al));
                ab = al+((float)0.618*(au-al));
                fa = funct(aa, n);
                fb = funct(ab, n);
                break;
            default:
                break;
        }
    }

    *a = (au+al)/(float)2;      //send minimum alpha value to pointer address
    *f = funct(*a, n);          //send minimum function value to pointer address
}

/* calculates the function value
    a = value of alpha, input
    fVal = function value on return
    nC = number of calls for function evaluation
**/
float funct(float a, int *nC)
{
    float fVal = 0;
    *nC = *nC + 1;                  //send function evaluation count to pointer
    address
    fVal = 4.1*pow(a,2)-5*a-6.5;     //enter function of one variable here
    return fVal;
}
```

Output: Initial lower alpha bound = 0.261796;
Initial upper alpha bound = 0.816263;
Minimum (α*) after two iterations = 0.579473;
Minimum function value (f*) = -8.02063;
No. of function evaluations = 12.

10.42 ———————————————————————————

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2;\ \ \mathbf{d} = [-4, -6]\ \ at\ \ \mathbf{x} = [4, 4]$$

Solution:

$f(\alpha) = 52\alpha^2 - 52\alpha + 13$ (from the solution to Problem 10.22)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 0.261796;
          Initial upper alpha bound = 0.816263;
          Minimum (α*) after two iterations = 0.498574;
          Minimum function value (f*) = 0.000105676;
          No. of function evaluations = 12.
```

10.43 ———————————————————————————

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = 10(x_2 - x_1^2)^2 + (1 - x_1)^2;\ \mathbf{d} = [-162, 40]\ \text{at}\ \mathbf{x} = [2, 2]$$

Solution:

$f(\alpha) = 6887475360\alpha^4 - 361117440\alpha^3 + 5809444\alpha^2 - 27844\alpha + 41$

(from the solution to Problem 10.23)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 0;
          Initial upper alpha bound = 0.1309;
          Minimum (α*) after two iterations = 0.025;
          Minimum function value (f*) = 23.7626;
          No. of function evaluations = 6.
```

10.44 ———————————————————————————

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = (x_1 - 2)^2 + x_2^2;\ \mathbf{d} = [2, -2]\ \text{at}\ \mathbf{x} = [1, 1]$$

Solution:

$f(\alpha) = 8\alpha^2 - 8\alpha + 2$ (from the solution to Problem 10.24)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 0.261796;
          Initial upper alpha bound = 0.816263;
          Minimum (α*) after two iterations = 0.498574;
          Minimum function value (f*) = 0.0000162579;
          No. of function evaluations = 12.
```

10.45

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = 0.5x_1^2 + x_2^2 - x_1x_2 - 7x_1 - 7x_2;\ \mathbf{d} = [7, 6] \text{ at } \mathbf{x} = [1,1]$$

Solution:

$f(\alpha) = 18.5\alpha^2 - 85\alpha - 13.5$ (from the solution to Problem 10.25)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 1.37071;
          Initial upper alpha bound = 3.71932;
          Minimum (α*) after two iterations = 2.37366;
          Minimum function value (f*) = -111.027;
          No. of function evaluations = 18.
```

10.46

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = (x_1 + x_2)^2 + (x_2 + x_3)^2;\ \mathbf{d} = [-4, -8, -4] \text{ at } \mathbf{x} = [1,1,1]$$

Solution:

$f(\alpha) = 288\alpha^2 - 96\alpha + 8$ (from the solution to Problem 10.26)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 0.05;
          Initial upper alpha bound = 0.261796;
          Minimum (α*) after two iterations = 0.171347;
          Minimum function value (f*) = 0.00630861;
          No. of function evaluations = 8.
```

10.47

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2;\ \mathbf{d} = [-2, -4, 2] \text{ at } \mathbf{x} = [1, 2, -1]$$

Solution:

$f(\alpha) = 24\alpha^2 - 24\alpha + 6$ (from the solution to Problem 10.27)

Solved using a modified form of the program in Section B.2, Appendix B.

```
Output:   Initial lower alpha bound = 0.261796;
          Initial upper alpha bound = 0.816263;
          Minimum (α*) after two iterations = 0.498574;
          Minimum function value (f*) = 0.0000487736;
          No. of function evaluations = 12.
```

10.48

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = (x_1 + 3x_2 + x_3)^2 + 4(x_1 - x_2)^2;\ \mathbf{d} = [1,3,1] \text{ at } \mathbf{x} = [-1,-1,-1]$$

Solution:

$f(\alpha) = 137\alpha^2 - 110\alpha + 25$ (from the solution to Problem 10.28)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 0.261796;
Initial upper alpha bound = 0.816263;
Minimum (α*) after two iterations = 0.367691;
Minimum function value (*f**) = 3.07593;
No. of function evaluations = 12.

10.49

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = 9 - 8x_1 - 6x_2 - 4x_3 + 2x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_2x_3;\ \mathbf{d} = [2,-2,0] \text{ at } \mathbf{x} = [1,1,1]$$

Solution:

$f(\alpha) = 8\alpha^2 - 8\alpha$ (from the solution to Problem 10.29)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 0.261796;
Initial upper alpha bound = 0.816263;
Minimum (α*) after two iterations = 0.498574;
Minimum function value (*f**) = -1.99998;
No. of function evaluations = 12.

10.50

For the following function, calculate the initial interval of uncertainty for the golden section search with δ=0.05 at the given point and the search direction; then complete two iterations of the Phase II of the method.

$$f(\mathbf{x}) = (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2 + (x_4 - 4)^2;\ \mathbf{d} = [-2,2,-2,2] \text{ at } \mathbf{x} = [2,1,4,3]$$

Solution:

$f(\alpha) = 16\alpha^2 - 16\alpha + 4$ (from the solution to Problem 10.30)

Solved using a modified form of the program in Section B.2, Appendix B.

Output: Initial lower alpha bound = 0.261796;
Initial upper alpha bound = 0.816263;
Minimum (α*) after two iterations = 0.498574;
Minimum function value (*f**) = 0.0000325157;
No. of function evaluations = 12.

Section 10.6 Search Direction Determination: Steepest Descent Method

10.51

Answer True or False.

Solution:

1. The steepest-descent method is convergent. *True*
2. The steepest-descent method can converge to a local maximum point starting from a point where the gradient of the function is nonzero. *False*
3. Steepest-descent directions are orthogonal to each other. *True*
4. Steepest-descent direction is orthogonal to the cost surface. *True*

10.52

For the following function, complete two iterations of the steepest-descent method starting from the given design point.

$$f(x_1, x_2) = x_1^2 + 2x_2^2 - 4x_1 - 2x_1x_2; \text{ starting point, } \mathbf{x} = [1,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [2x_1 - 4 - 2x_2, 4x_2 - 2x_1] = [-4, 2], \left\|\mathbf{c}^{(0)}\right\| = \sqrt{20} > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [4, -2]$
4. Calculate a step size α:

   $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1,1] + \alpha[4,-2] = [1+4\alpha, 1-2\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 40\alpha^2 - 20\alpha - 3$

   $f'(\alpha) = 80\alpha - 20 = 0, \alpha_0 = 0.25$

   $f''(\alpha_0) = 80 > 0$, So α_0 = 0.25 is a minimum point.
5. Update the design:

   $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [2, 0.5], k = k+1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [2, 0.5], k = 1, \varepsilon = 0.001$
2. $\mathbf{c}^{(1)} = [-1, -2], \left\|\mathbf{c}^{(1)}\right\| = \sqrt{5} > \varepsilon$
3. $\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} = [1, 2]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [2, 0.5] + \alpha[1,2] = [2+\alpha, 0.5+2\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 5\alpha^2 - 5\alpha - 5.5$

   $f'(\alpha) = 10\alpha - 5 = 0, \alpha_1 = 0.5$

   $f''(\alpha_1) = 10 > 0$, So, α_1 = 0.5 is a minimum point.
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = [2.5, 1.5], k = k+1$

10.53 ———

For the following function, complete two iterations of the steepest-descent method starting from the given design point.

$$f(x_1, x_2) = 12.096x_1^2 + 21.504\,x_2^2 - 1.7321x_1 - x_2;\ \text{starting point, } \mathbf{x} = [1,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [24.192x_1 - 1.7321, 43.008\,x_2 - 1] = [22.4599, 42.008], \left\|\mathbf{c}^{(0)}\right\| = 47.6 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-22.4599, -42.008]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1 - 22.4599\alpha, 1 - 42.008\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 12.096(1 - 22.4599\alpha)^2 + 21.504(1 - 42.008\alpha)^2 - 1.7321(1 - 22.4599\alpha)$
   $-(1 - 42.008\alpha)$

   $f'(\alpha) = -2269.1192 + 88098.6\alpha = 0, \alpha_0 = 0.025757$

   $f''(\alpha_0) = 88098.6 > 0$, So, α_0 = 0.025757 is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.42151, -0.08198], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.42151, -0.08198], k = 1$
2. $\mathbf{c}^{(1)} = [8.465, -4.525], \left\|\mathbf{c}^{(1)}\right\| = 9.6 > \varepsilon$
3. $\mathbf{d}^{(1)} = -\mathbf{c}^{1} = [-8.465, 4.525]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.4215 - 8.465\alpha, -0.082 + 4.525\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 12.096(0.4215 - 8.465\alpha)^2 + 21.504(-0.082 + 4.525\alpha)^2$
   $-1.7321(0.4215 - 8.465\alpha) - (-0.082 + 4.525\alpha)$

   $f'(\alpha) = 2614.123\alpha - 92.138 = 0, \alpha_1 = 0.035246$

   $f''(\alpha_1) = 2614 > 0$, So α_1 = 0.035246 is a minimum point.
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = [0.1231, 0.0775], k = k + 1$

10.54

For the following function, complete two iterations of the steepest-descent method starting from the given design point.

$$f(x_1, x_2) = 6.983x_1^2 + 12.415x_2^2 - x_1; \text{ starting point, } \mathbf{x} = [2,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [2,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [13.966x_1 - 1, 24.83\,\mathrm{x}_2] = [26.932, 24.83], \left\|\mathbf{c}^{(0)}\right\| = 36.6314 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-26.932, -24.83]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [2 - 26.932\alpha, 1 - 24.83\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 6.983(2 - 26.932\alpha)^2 + 12.415(1 - 24.83\alpha)^2 - (2 - 26.936\alpha)$

   $f\,'(\alpha) = 25441.4173\alpha - 1341.97325 = 0, \alpha_0 = 0.05275$

   $f\,''(\alpha_0) = 25441 > 0;$ So, α_0 = 0.05275 is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.579, -0.310], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.579, -0.310], k = 1$
2. $\mathbf{c}^{(1)} = [7.086, -7.697], \left\|\mathbf{c}^{(1)}\right\| = 10.4621 > \varepsilon$
3. $\mathbf{d}^{(1)} = -\mathbf{c}^{1} = [-7.086, 7.697]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.579 - 7.086\alpha, -0.310 + 7.697\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 6.983(0.579 - 7.086\alpha)^2 + 12.415(-0.31 + 7.697\alpha)^2 - (0.579 - 7.086\alpha)$

   $f\,'(\alpha) = 2172.276\alpha - 109.460 = 0, \alpha_1 = 0.0504$

   $f\,''(\alpha_1) = 2172 > 0,$ So α_1 = 0.0504 is a minimum point.
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = [0.222, 0.0778], k = k + 1$

10.55

For the following function, complete two iterations of the steepest-descent method starting from the given design point.

$f(x_1, x_2) = 12.096x_1^2 + 21.504x_2^2 - x_2$; starting point, $\mathbf{x} = [1, 2]$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1, 2], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [24.192x_1, 43.008\,\mathrm{x}_2 - 1] = [24.192, 85.016], \left\|\mathbf{c}^{(0)}\right\| = 88.391 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-24.192, -85.016]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1 - 24.192\alpha, 2 - 85.016\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 12.096(1 - 24.192\alpha)^2 + 21.504(2 - 85.016\alpha)^2 - (2 - 85.016\alpha)$

   $f'(\alpha) = 325008.23\alpha - 7812.973 = 0, \alpha_0 = 0.02404$

   $f''(\alpha_0) = 325008 > 0,$ So, α_0 = 0.02404 is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.418, -0.0437], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.418, -0.0437], k = 1$
2. $\mathbf{c}^{(1)} = [10.112, -2.879], \left\|\mathbf{c}^{(1)}\right\| = 10.514 > \varepsilon$
3. $\mathbf{d}^{(1)} = -\mathbf{c}^{1} = [-10.112, 2.879]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.418 - 10.112\alpha, -0.0437 + 2.879\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 12.096(0.418 - 10.112\alpha)^2 + 21.504(-0.0437 + 2.879\alpha)^2$

   $-(-0.0437 + 2.879\alpha)$

   $f'(\alpha) = 2830.171\alpha - 110.545 = 0, \alpha_1 = 0.03906$

   $f''(\alpha_1) = 2830 > 0,$ So α_1 = 0.03906 is a minimum point.
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = [0.0230, 0.0688], k = k + 1$

10.56

For the following function, complete two iterations of the steepest-descent method starting from the given design point.

$f(x_1, x_2) = 25x_1^2 + 20x_2^2 - 2x_1 - \mathrm{x}_2;$ starting point, $\mathbf{x} = [3,1]$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [3,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [50x_1 - 2, 40\,\mathrm{x}_2 - 1] = [148, 39], \left\|\mathbf{c}^{(0)}\right\| = 153.05 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-148, -39]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [3 - 148\alpha, 1 - 39\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 25(3 - 148\alpha)^2 + 20(1 - 39\alpha)^2 - 2(3 - 148\alpha) - (1 - 39\alpha)$

   $f\,'(\alpha) = 1156040\alpha - 23425 = 0, \alpha_0 = 0.02026$

   $f\,''(\alpha_0) = 1156040 > 0;$ So, α_0 = 0.02026 is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.001055, 0.2097], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.001055, 0.2097], k = 1$
2. $\mathbf{c}^{(1)} = [-1.947, 7.388], \left\|\mathbf{c}^{(1)}\right\| = 7.64 > \varepsilon$
3. $\mathbf{d}^{(1)} = -\mathbf{c}^{1} = [1.947, -7.388]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.001055 + 1.947\alpha, 0.2097 - 7.388\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 25(0.001055 + 1.947\alpha)^2 + 20(0.2097 - 7.388\alpha)^2 - 2(0.001055 + 1.947\alpha)$
   $-(0.2097 - 7.388\alpha)$

   $f\,'(\alpha) = 2372.842\alpha - 58.374 = 0, \alpha_1 = 0.0246$

   $f\,''(\alpha_1) = 2373 > 0,$ So α_1 = 0.0246 is a minimum point.
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = [0.0490, 0.0280], k = k + 1$

10.57 ——

For the following function, complete two iterations of the steepest-descent method starting from the given design point.

$$f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3;\ \text{starting point, } \mathbf{x} = [1,1,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1,1,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [2x_1 + 2x_2, 4x_2 + 2x_1 + 2x_3, 4x_3 + 2x_2] = [4,8,6], \|\mathbf{c}^{(0)}\| = 10.77 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-4,-8,-6]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1-4\alpha, 1-8\alpha, 1-6\alpha]$
   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = (1-4\alpha)^2 + 2(1-8\alpha)^2 + 2(1-6\alpha)^2 + 2(1-4\alpha)(1-8\alpha) + 2(1-8\alpha)(1-6\alpha)$
   $f'(\alpha) = 752\alpha - 116 = 0, \alpha_0 = 0.154255$
   $f''(\alpha_0) = 752 > 0;$ So, α_0 = 0.154255 is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.383, -0.234, 0.0745], k = k+1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.383, -0.234, 0.0745], k = 1$
2. $\mathbf{c}^{(1)} = [0.298, -0.021, -0.17], \|\mathbf{c}^{(1)}\| = 0.3437 > \varepsilon$
3. $\mathbf{d}^{(1)} = -\mathbf{c}^{1} = [-0.298, 0.021, 0.17]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.383 - 0.298\alpha, -0.234 + 0.021\alpha, 0.0745 + 0.17\alpha]$
   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = (0.383 - 0.298\alpha)^2 + 2(-0.234 + 0.021\alpha)^2 + 2(0.0745 + 0.17\alpha)^2$
   $+2(0.383 - 0.298\alpha)(-0.234 + 0.021\alpha) + 2(-0.234 + 0.021\alpha)(0.0745 + 0.17\alpha)$
   $f'(\alpha) = 0.28422\alpha - 0.118145 = 0, \alpha_1 = 0.4157$
   $f''(\alpha_1) = 0.28422 > 0,$ So α_1 = 0.4157 is a minimum point.
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = [0.259, -0.225, 0.145], k = k+1$

10.58

For the following function, complete two iterations of the steepest-descent method starting from the given design point. The step size may be approximated or calculated using a computer program.

$$f(x_1,x_2)=8x_1^2+8x_2^2-80\sqrt{x_1^2+x_2^2-20x_2+100}-80\sqrt{x_1^2+x_2^2+20x_2+100}-5\,\mathrm{x}_1-5\,\mathrm{x}_2;$$

starting point, $\mathbf{x}=[4,6]$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)}=[4,6], k=0, \varepsilon=0.001$
2. $\mathbf{c}^{(0)}=[-16.9714, 69.9571], \left\|\mathbf{c}^{(0)}\right\|=71.9863>\varepsilon$
3. $\mathbf{d}^{(0)}=-\mathbf{c}^{(0)}=[16.9714,-69.9571]$
4. $\mathbf{x}^{(1)}=\mathbf{x}^{(0)}+\alpha\mathbf{d}^{(0)}=[4+16.9714\alpha, 6-69.9571\alpha]$

$$f(\alpha)=f(\mathbf{x}^{(0)}+\alpha\mathbf{d}^{(0)})=8(4+16.9714\alpha)^2+8(6-69.9571\alpha)^2-5(10-52.9857\alpha)$$
$$-80\sqrt{(4+16.9714\alpha)^2+(6-69.9571\alpha)^2-20(6-69.9571\alpha)+100}$$
$$-80\sqrt{(4+16.9714\alpha)^2+(6-69.9571\alpha)^2+20(6-69.9571\alpha)+100}$$

(Using the Golden Section Method in Appendix B to solve for α_0); $\alpha_0=0.07885$

5. $\mathbf{x}^{(1)}=\mathbf{x}^{(0)}+\alpha_0\mathbf{d}^{(0)}=[5.3382, 0.4839], k=k+1$

Iteration 2:

1. $\mathbf{x}^{(1)}=[5.3382, 0.4839], k=1$
2. $\mathbf{c}^{(1)}=[4.9720, 1.2237], \left\|\mathbf{c}^{(1)}\right\|=5.124>\varepsilon$
3. $\mathbf{d}^{(1)}=-\mathbf{c}^{1}=[-4.9720,-1.2237]$
4. $\mathbf{x}^{(2)}=\mathbf{x}^{(1)}+\alpha\mathbf{d}^{(1)}=[5.3382-4.9720\alpha, 0.4839-1.2237\alpha]$

$$f(\alpha)=f(\mathbf{x}^{(1)}+\alpha\mathbf{d}^{(1)})=8(5.3382-4.9720\alpha)^2+8(0.4839-1.2237\alpha)^2-5(5.8221-6.1957\alpha)$$
$$-80\sqrt{(5.3382-4.9720\alpha)^2+(0.4839-1.2237\alpha)^2-20(0.4839-1.2237\alpha)+100}$$
$$-80\sqrt{(5.3382-4.9720\alpha)^2+(0.4839-1.2237\alpha)^2+20(0.4839-1.2237\alpha)+100}$$

(Using the Golden Section Method in Appendix B to solve for α_1); $\alpha_1=0.2141$

5. $\mathbf{x}^{(2)}=\mathbf{x}^{(1)}+\alpha_1\mathbf{d}^{(1)}=[4.2737, 0.2219], k=k+1$

10.59

For the following function, complete two iterations of the steepest-descent method starting from the given design point. The step size may be approximated or calculated using a computer program.

$$f(x_1,x_2)=9x_1^2+9x_2^2-100\sqrt{x_1^2+x_2^2-20x_2+100}-64\sqrt{x_1^2+x_2^2+16x_2+64}-5x_1-41x_2;$$

starting point, $\mathbf{x}=[5,2]$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)}=[5,2], k=0, \varepsilon=0.001$
2. $\mathbf{c}^{(0)}=[3.378436, 22.55649], \|\mathbf{c}^{(0)}\|=22.8>\varepsilon$
3. $\mathbf{d}^{(0)}=-\mathbf{c}^{(0)}=[-3.378436,-22.55649]$
4. $\mathbf{x}^{(1)}=\mathbf{x}^{(0)}+\alpha\mathbf{d}^{(0)}=[5-3.378436\alpha, 2-22.55649\alpha]$

$$f(\alpha)=f(\mathbf{x}^{(0)}+\alpha\mathbf{d}^{(0)})=9(5-3.378436\alpha)^2+9(2-22.55649\alpha)^2-5(5-3.378436\alpha)$$
$$-41(2-22.55649\alpha)-100\sqrt{(5-3.378436\alpha)^2+(2-22.55649\alpha)^2-20(2-22.55649\alpha)+100}$$
$$-64\sqrt{(5-3.378436\alpha)^2+(2-22.55649\alpha)^2+16(2-22.55649\alpha)+64}$$

(Using the Golden Section Method in Appendix B to solve for α_0); $\alpha_0=0.07327$

5. $\mathbf{x}^{(1)}=\mathbf{x}^{(0)}+\alpha_0\mathbf{d}^{(0)}=[4.75246, 0.34729], k=k+1$

Iteration 2:

1. $\mathbf{x}^{(1)}=[4.75246, 0.34729], k=1$
2. $\mathbf{c}^{(1)}=[4.7079,-0.6505], \|\mathbf{c}^{(1)}\|=4.75>\varepsilon$
3. $\mathbf{d}^{(1)}=-\mathbf{c}^{1}=[-4.7079, 0.6505]$
4. $\mathbf{x}^{(2)}=\mathbf{x}^{(1)}+\alpha\mathbf{d}^{(1)}=[4.75246-4.7079\alpha, 0.34729+0.6505\alpha]$

$$f(\alpha)=f(\mathbf{x}^{(1)}+\alpha\mathbf{d}^{(1)})=9(4.75246-4.7079\alpha)^2+9(0.34729+0.6505\alpha)^2-5(4.75246-4.7079\alpha)$$
$$-41(0.34729+0.6505\alpha)$$
$$-100\sqrt{(4.75246-4.7079\alpha)^2+(0.34729+0.6505\alpha)^2-20(0.34729+0.6505\alpha)+100}$$
$$-64\sqrt{(4.75246-4.7079\alpha)^2+(0.34729+0.6505\alpha)^2+16(0.34729+0.6505\alpha)+64}$$

(Using the Golden Section Method in Appendix B to solve for α_1); $\alpha_1=0.19582$

5. $\mathbf{x}^{(2)}=\mathbf{x}^{(1)}+\alpha_1\mathbf{d}^{(1)}=[3.8306, 0.4747], k=k+1$

10.60

For the following function, complete two iterations of the steepest-descent method starting from the given design point.

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2;\ \text{starting point, } \mathbf{x} = [5, 2]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [5, 2], k = 0, \varepsilon = 0.00001$
2. $\mathbf{c}^{(0)} = [x_1\left(400x_1^2 + 2\right) - \left(400x_1x_2\right) - 2, 200x_2 - 200x_1^2] = [46008, -4600], \left\|\mathbf{c}^{(0)}\right\| = 46237 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-46008, 4600]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [5 - 46008\alpha, 2 + 4600\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 100\left((2 + 4600\alpha) - (5 - 46008\alpha)^2\right)^2 + \left(1 - (5 - 46008\alpha)\right)^2$

   (Using the Golden Section Method in Appendix B to solve for α_0); $\alpha_0 = 0.00014$
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [-1.44, 2.644], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [-1.44, 2.644], k = 1$
2. $\mathbf{c}^{(1)} = [323.67, 114.08], \left\|\mathbf{c}^{(1)}\right\| = 343.19 > \varepsilon$
3. $\mathbf{d}^{(1)} = -\mathbf{c}^{1} = [-323.67, -114.08]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [-1.44 - 323.67\alpha, 2.644 - 114.08\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 100\left((2.644 - 114.08\alpha) - (-1.44 - 323.67\alpha)^2\right)^2 + (1.44 + 323.67\alpha)^2$

   (Using the Golden Section Method in Appendix B to solve for α_1); $\alpha_1 = 0.000516$
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = [-1.607, 2.585], k = k + 1$

10.61

For the following function, complete two iterations of the steepest-descent method starting from the given design point.

$$f(x_1, x_2, x_3, x_4) = (x_1 - 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4; \text{ starting point, } \mathbf{x} = [1, 2, 3, 4]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1, 2, 3, 4], k = 0, \varepsilon = 0.00001$
2. $\mathbf{c}^{(0)} = [2(x_1 - 10x_2) + 40(x_1 - x_4)^3, -20(x_1 - 10x_2) + 4(x_2 - 2x_3)^3, 10(x_3 - x_4) - 8(x_2 - 2x_3)^3,$ $-10(x_3 - x_4) - 40(x_1 - x_4)^3] = [-1118, 124, 502, 1090], \left\|\mathbf{c}^{(0)}\right\| = 1645 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [1118, -124, -502, -1090]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1 + 1118\alpha, 2 - 124\alpha, 3 - 502\alpha, 4 - 1090\alpha]$
   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = (2358\alpha - 19)^2 + 5(588\alpha - 1)^2 + (880\alpha - 4)^4 + 10(2208\alpha - 3)^4$
   (Using the Golden Section Method in Appendix B to solve for α_0); $\alpha_0 = 0.001854$
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [3.0728, 1.7701, 2.06929, 1.97914], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [3.0728, 1.7701, 2.06929, 1.97914], k = 1$
2. $\mathbf{c}^{(1)} = [23.068, 239.418, 107.193, -53.226], \left\|\mathbf{c}^{(1)}\right\| = 268.66 > \varepsilon$
3. $\mathbf{d}^{(1)} = -\mathbf{c}^{1} = [-23.068, -239.418, -107.193, 53.226]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [3.0728 - 23.068\alpha, 1.7701 - 239.418\alpha, 2.06929 - 107.193\alpha, 1.97914$ $+ 53.226\alpha]$
   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = (-14.63 + 2371.11\alpha)^2 + 5(0.0902 - 160.42\alpha)^2 + (-2.3685 - 25.032\alpha)^4$ $+ 10(1.0937 - 76.294\alpha)^4$
   (Using the Golden Section Method in Appendix B to solve for α_1); $\alpha_1 = 0.00597616$
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = [2.9349, 0.3393, 1.4287, 2.2972], k = k + 1$

10.62

Solve Exercises 10.52 to 10.61 using the computer program given in Appendix B for the steepest-descent method.

Solutions:

The solutions were obtained using the following program, written in C++.

```
#include <iostream>
#include <math.h>

using namespace std;

void goldenIntervalSearch(float x[], float xn[], float d[], float *a, float delta,
    float epsilon, float *f, int *nCount, int ndv);
float funct(float xVector[], int *nC);
void grad(float xVec[], float grad[]);
void scale(float arrOne[], float arrTwo[], int scalar, int n);
float tNorm(float vec[], int n);
void update(float x[], float xwrk[], float d[], float a, int n);
void printStuff(int currCount, float x[], float alph, float grad[], float f, int
    ndv);

int main()
{
    float delta = 0.1;        // * modify this value for the required delta for
    line search
    float epsilon = 0.001;    // * modify this value for the required line search
    accuracy
    float epsilon2 = 0.001;   // * modify this value for steepest-descent method
    stopping criterion
    int ndv = 2;              // * modify this value for number of design variables
    int nCount = 0;           //   tracks number of function evaluations
    int noc = 100;        // * modify this value for number of cycles of the method
    int nocActual = 0;    //   tracks number of cycles of the method

    float f = 0;              //   holds the value of the function at the minimum
    float x[2] = {1, 1};      // * modify this array to hold design variable vector
    float d[2] = {0, 0};        // * modify this array to hold the direction vector
    float c[2] = {0, 0};        //   holds the gradient vector
    float wrk[2] = {0, 0};      //   vector to hold temporary values
    float alpha = 0;            //   holds the value of alpha at the minimum
    float temp = 0;

    float *alphaptr = &alpha;   //  pointer to alpha
    float *fptr = &f;           //  pointer to f
    int *nptr = &nCount;        //  pointer to nCount

    for(int k = 1; k <= noc; k++)
    {
        f = funct(x, nptr);
        grad(x, c);
        temp = tNorm(c, ndv);
        printStuff(k, x, alpha, c, f, ndv);
        if(temp <= epsilon2)
        {
            k = noc+1;
            for(int l = 0; l < ndv; l++)
            {
                cout<<"x["<<l<<"] = "<<x[l]<<"\n";
            }
        }
        else if(temp > epsilon2)
        {
            scale(c, d, -1, ndv);
            goldenIntervalSearch(x, wrk, d, alphaptr, delta, epsilon, fptr, nptr,
    ndv);
            update(x, x, d, alpha, ndv);
            nocActual = k;
        }
    }

    if(nocActual >= noc)
    {
        cout<<"Limit on number of cycles has exceeded maximum."<<"\n";
        for(int l = 0; l < ndv; l++)
        {
            cout<<"x["<<l<<"] = "<<x[l]<<"\n";
        }
    }

    cout <<"f* = "<<f<<"\n";               //outputs optimum cost function value
    cout <<"NFE: "<<nCount<<"\n";     //outputs the number of function evaluations
    cout<<"NIT: "<<nocActual;

    return 0;
}

/* this function implements golden interval search
    a = optimal value of alpha on return
    d = initial step length
    e = convergence parameter
    f = optimum value of the function on return
    n = number of function evaluations on return
**/
void goldenIntervalSearch(float x[], float xn[], float d[], float *a, float delta,
    float epsilon, float *f, int *nCount, int ndv)
{
    int q = 1;                          //delta multiplier
    float gr = 0.5*pow(5,0.5)+0.5;      //holds value of golden section multiplier
    float delta1 = delta;               //holds optimal value of delta, initially
    set to passed value
    float al = 0;                       //lower bound of alpha
    float aa = delta1;                  //holds value of mid-point alpha a,
    initially set to delta1
    float ab = 0;                       //holds value of mid-point alpha b
    float au = aa+(pow(gr, q)*delta1);  //holds value of upper bound of alpha
    float fl = 0;
    float fa = 0;                      //holds function value of mid-point alpha
    float fb = 0;                      //holds function value at mid-point alpha b
    float fu = 0;                      //holds function value of upper alpha bound
```

```
update(x, xn, d, al, ndv);   //call function update() to update the value of xn
fl = funct(xn, nCount);      //call function funct() to return the value at a
given value of alpha
update(x, xn, d, aa, ndv);
fa = funct(xn, nCount);
update(x, xn, d, au, ndv);
fu = funct(xn, nCount);

//establish initial interval of uncertainty
while(fa > fu)
{
    q = q+1;
    al = aa;
    aa = au;
    au = au+(pow((float)1.618, q)*delta1);
    fl = fa;
    fa = fu;
    update(x, xn, d, au, ndv);
    fu = funct(xn, nCount);
}

ab = al+((au-al)/gr);
update(x, xn, d, ab, ndv);
fb = funct(xn, nCount);

//refine the interval of uncertainty further
while((au-al) >= epsilon)
{
    int caseInPoint = 0;

    if(fa < fb)
        caseInPoint = 1;
    else if(fa > fb)
        caseInPoint = 2;
    else if(fa == fb)
        caseInPoint = 3;

    switch(caseInPoint)
    {
        case 1:
            au = ab;
            ab = aa;
            fb = fa;
            fu = fb;
            aa = al+((1-1/gr)*(au-al));
            update(x, xn, d, aa, ndv);
            fa = funct(xn, nCount);
            break;
        case 2:
            al = aa;
            aa = ab;
            fl = fa;
            fa = fb;
            ab = al+((au-al)/gr);
            update(x, xn, d, ab, ndv);
            fb = funct(xn, nCount);
```

```
                break;
            case 3:
                al = aa;
                au = ab;
                fl = fa;
                fu = fb;
                aa = al+((1-1/gr)*(au-al));
                update(x, xn, d, aa, ndv);
                fa = funct(xn, nCount);
                ab = al+((au-al)/gr);
                update(x, xn, d, ab, ndv);
                fb = funct(xn, nCount);
                break;
            default:
                break;
        }

    }

    *a = (au+al)/(float)2;      //send minimum alpha value to pointer address
}

/* calculates the function value
    a = value of alpha, input
    fVal = function value on return
    nC = number of calls for function evaluation
**/
float funct(float xVector[], int *nC)
{
    float fVal = 0;
    float x1 = xVector[0];
    float x2 = xVector[1];
    *nC = *nC + 1;                  //send function evaluation count to pointer
    address
    fVal = pow(x1, 2)+2*pow(x2, 2)-4*x1-2*x1*x2;     //enter function here

    return fVal;
}

void grad(float xVec[], float grad[])
{
    float x1 = xVec[0];
    float x2 = xVec[1];
    grad[0] = (2*x1)-4-(2*x2);
    grad[1] = (4*x2)-(2*x1);
}

void scale(float arrOne[], float arrTwo[], int scalar, int n)
{
    for(int i = 0; i < n; i++)
    {
        arrTwo[i] = scalar*arrOne[i];
    }
}

float tNorm(float vec[], int n)
{
    float sum = 0;
    float tNorm = 0;

    for(int i = 0; i < n; i++)
    {
        sum = sum+pow(vec[i], 2);
    }

    tNorm = pow(sum, 0.5);

    return tNorm;
}


void update(float x[], float xwrk[], float d[], float a, int n)
{
    for(int i = 0; i < n; i++)
    {
        xwrk[i] = x[i]+(a*d[i]);
    }
}

void printStuff(int currCount, float x[], float alph, float grad[], float f, int
    ndv)
{
    cout<<"Iteration number: "<<currCount<<" Function value: "<<f<<" Alpha value:
    "<<alph<<"\n";
    cout<<"Norm of gradient vector: "<<tNorm(grad, ndv)<<"\n";
}
```

10.52: **x*** = (3.999, 1.99936), f * = -8, Number of Iterations (NIT) = 25, Number of Function Evaluations (NFE) = 452;
10.53: **x*** = (0.0716089, 0.0232635), f * = − 0.0736332, NIT = 8, NFE = 136;
10.54: **x*** = (0.0716167, − 0.0000150918), f * = − 0.0358012, NIT = 9, NFE = 150;
10.55: **x*** = (0.000010756, 0.0232737), f * = − 0.0116257, NIT = 6, NFE = 103;
10.56: **x*** = (0.0400012, 0.0250021), f * = − 0.0525, NIT = 5, NFE = 84;
10.57: **x*** = (0.00122397, − 0.00108302, 0.00049956), f * = 0.00000060985, NIT = 41, NFE = 795;
10.58: **x*** = (4.14465, 0.361625), f * = − 1616.18, NIT = 19, NFE = 265;
10.59: **x*** = (3.73284, 0.341954), f * = − 1526.56, NIT = 100, NFE = 1320;
10.60: **x*** = (0.998932, 0.88786), f * = 0.00000114375, NIT = 315, NFE = 5078;
10.61: **x*** = (0.129072, 0.0129158, 0.0692521, 0.0704564), f * = 0.000374958, NIT = 1000, NFE = 17000;

10.63

Consider the following three functions. Minimize f₁, f₂, *and* f₃ *using the program for the steepest-descent method given in Appendix B. Choose the starting design to be* $\mathbf{x} = [1, 1, 2]$ *for all functions. What do you conclude from observing the performance of the method on the foregoing functions?*

$$f_1(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2;\; f_2 = x_1^2 + 10x_2^2 + 100x_3^2;\; f_3 = 100x_1^2 + x_2^2 + 0.1x_3^2$$

Solutions:

For f_1: **x*** = (0.000198483, 0.000198483, 0.000396967), f_1^* = 2.36374E−07, NIT = 1, NFE = 23;

For f_2: **x*** = (0.000407165, 0.0, 0.00000281032), f_2^* = 0.000000166573, NIT = 345, NFE = 5522;

For f_3: **x*** = (-0.00000143735, 0.0, 0.00476033), f_3^* = 0.00000226628, NIT = 998, NFE = 15984.

For f_1, just one iteration is needed for the solution, while for f_2 and f_3, 345 and 998 iterations are needed, respectively. The reason is that the condition number of the Hessian of f_1 is one, and that of f_2 and f_3 is 100 and 1000 respectively.

To get the solutions in one iteration for f_2 and f_3, the following transformations are suggested. (With these, the condition number becomes one):

For f_2: $\mathbf{H} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 200 \end{bmatrix}$, $\mathbf{x} = \begin{bmatrix} 1/\sqrt{2} & 0 & 0 \\ 0 & 1/\sqrt{20} & 0 \\ 0 & 0 & 1/\sqrt{200} \end{bmatrix}\mathbf{y}$

For f_3: $\mathbf{H} = \begin{bmatrix} 200 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}$, $\mathbf{x} = \begin{bmatrix} 1/\sqrt{200} & 0 & 0 \\ 0 & 1/\sqrt{2} & 0 \\ 0 & 0 & 1/\sqrt{0.2} \end{bmatrix}\mathbf{y}$

10.64

Calculate the gradient of the following functions at the given points by the forward, backward, and central difference approaches with a 1 percent change in the point and compare them with the exact gradient.

$f_1(\mathbf{x}) = 12.096x_1^2 + 21.504x_2^2 - 1.7321x_1 - x_2$; starting point $\mathbf{x} = [5, 6]$

$f_2(\mathbf{x}) = 50(x_2 - \mathrm{x}_1^2)^2 + (2 - \mathrm{x}_1)^2$; starting point $\mathbf{x} = [1, 2]$

$f_3(\mathbf{x}) = x_1^2 + 2x_2^2 + 2x_3^2 + 2\,\mathrm{x}_1\mathrm{x}_2 + 2\,\mathrm{x}_2\mathrm{x}_3$; starting point $\mathbf{x} = [1, 2, 3]$

Solution:

(i.) $f_1(\mathbf{x}) = 12.096x_1^2 + 21.504x_2^2 - 1.7321x_1 - x_2$; starting point $\mathbf{x} = [5, 6]$

1. Exact: $\nabla f_1(\mathbf{x}) = [\partial f_1/\partial x_1, \partial f_1/\partial x_2] = [24.192x_1 - 1.7321, 43.008x_2 - 1] = [119.2, 258.0]$

2. Forward Difference Approach

$\partial f_1/\partial x_1 = [f_1(5.05, 6) - f_1(5, 6)]/0.05 = 5.9916/0.05 = 119.8$

$\partial f_1/\partial x_2 = [f_1(5, 6.06) - f_1(5, 6)]/0.06 = 15.50/0.06 = 258.3382$

3. Backward Difference Approach

$\partial f_1/\partial x_1 = [f_1(5, 6) - f_1(4.95, 6)]/0.05 = 5.9312/0.05 = 118.6$

$\partial f_1/\partial x_2 = [f_1(5, 6) - f_1(5, 5.94)]/0.06 = 15.3455/0.06 = 255.8$

4. Central Difference Approach

$\partial f_1/\partial x_1 = [f_1(5.05, 6) - f_1(4.95, 6)]/0.10 = 11.923/0.10 = 119.2$

$\partial f_1/\partial x_2 = [f_1(5, 6.06) - f_1(5, 5.94)]/0.12 = 30.846/0.12 = 258.0$

(ii.) $f_2(\mathbf{x}) = 50(x_2 - \mathrm{x}_1^2)^2 + (2 - \mathrm{x}_1)^2$; starting point $\mathbf{x} = [1, 2]$

1. Exact: $\nabla f_2(\mathbf{x}) = [\partial f_2/\partial x_1, \partial f_2/\partial x_2] = [100(x_2 - x_1^2)(-2\,\mathrm{x}_1) - 2(2 - \mathrm{x}_1), 100(\mathrm{x}_2 - \mathrm{x}_1^2)] = [-202, 100]$

2. Forward Difference Approach

$\partial f_2/\partial x_1 = [f_2(1.01, 2) - f_2(1, 2)]/0.01 = -2.0097/0.01 = -200.97$

$\partial f_2/\partial x_2 = [f_2(1, 2.02) - f_2(1, 2)]/0.02 = 2.02/0.02 = 101$

4. Backward Difference Approach

$\partial f_2/\partial x_1 = [f_2(1, 2) - f_2(0.99, 2)]/0.01 = -2.0299/0.01 = -202.99$

$\partial f_2/\partial x_2 = [f_2(1, 2) - f_2(1, 1.98)]/0.02 = 1.98/0.02 = 99$

4. Central Difference Approach

$\partial f_2/\partial x_1 = [f_2(1.01, 2) - f_2(0.99, 2)]/0.02 = -4.0396/0.02 = -201.98$

$\partial f_2/\partial x_2 = [f_2(1, 2.02) - f_2(1, 1.98)]/0.04 = 4/0.04 = 100$

(iii.) $f_3(\mathbf{x}) = x_1^2 + 2x_2^2 + 2x_3^2 + 2\,x_1x_2 + 2\,x_2x_3$; starting point $\mathbf{x} = [1, 2, 3]$

1. Exact: $\nabla f_3(\mathbf{x}) = [\partial f_3/\partial x_1, \partial f_3/\partial x_2] = [2x_1 + 2\,x_2, 4\,x_2 + 2\,x_1 + 2\,x_3, 4\,x_3 + 2\,x_2] = [6, 16, 16]$

2. Forward Difference Approach

$\partial f_3/\partial x_1 = [f_3(1.01, 2, 3) - f_3(1, 2, 3)]/0.01 = 0.0601/0.01 = 6.01$

$\partial f_3/\partial x_2 = [f_3(1, 2.02, 3) - f_3(1, 2, 3)]/0.02 = 0.3208/0.02 = 16.04$

$\partial f_3/\partial x_3 = [f_3(1, 2, 3.03) - f_3(1, 2, 3)]/0.03 = 0.4818/0.03 = 16.06$

3. Backward Difference Approach

$\partial f_3/\partial x_1 = [f_3(1, 2, 3) - f_3(0.99, 2, 3)]/0.01 = 0.0599/0.01 = 5.99$

$\partial f_3/\partial x_2 = [f_3(1, 2, 3) - f_3(1, 1.98, 3)]/0.02 = 0.3192/0.02 = 15.96$

$\partial f_3/\partial x_3 = [f_3(1, 2, 3) - f_3(1, 2, 2.97)]/0.03 = 0.4782/0.03 = 15.94$

4. Central Difference Approach

$\partial f_3/\partial x_1 = [f_3(1.01, 2, 3) - f_3(0.99, 2, 3)]/0.02 = 0.12/0.02 = 6$

$\partial f_3/\partial x_2 = [f_3(1, 2.02, 3) - f_3(1, 1.98, 3)]/0.04 = 0.64/0.04 = 16$

$\partial f_3/\partial x_3 = [f_3(1, 2, 3.03) - f_3(1, 2, 2.97)]/0.06 = 0.96/0.06 = 16$

10.65

Consider the following optimization problem. Here, $\mathbf{u} = [u_1, u_2, ..., u_n]$ *are components of a unit vector. Solve this optimization problem and show that the* **u** *that maximizes the preceding objective function is indeed in the direction of the gradient* **c**.

$$\text{maximize } \sum_{i=1}^{n} u_i \frac{\partial f}{\partial x_i} = (\mathbf{c} \bullet \mathbf{u})$$

$$\text{subject to the constraint } \sum_{i=1}^{n} u_i^2 = 1$$

Solution:

Lagrangian: $L = -\sum u_i \dfrac{\partial f}{\partial x_i} + \nu\left(\sum u_i^2 - 1\right)$

Necessary conditions: $\dfrac{\partial L}{\partial u_i} = -\dfrac{\partial f}{\partial x_i} + 2\nu u_i = 0; u_i = \dfrac{1}{2\nu}\dfrac{\partial f}{\partial x_i}$ (1)

$\dfrac{\partial L}{\partial \nu} = \sum u_i^2 - 1 = 0$ (2)

From (1), we get: $u_i \dfrac{\partial f}{\partial x_i} = 2\nu u_i^2$

Summing up the above equation and using (2), we get: $2\nu = \sum u_i \dfrac{\partial f}{\partial x_i}$

Therefore, $\mathbf{u} = \mathbf{c}/2\nu$.

Section 10.7 Search Direction Determination: Conjugate Gradient Method

10.66____

Answer True or False.

Solution:

1. The conjugate gradient method usually converges faster than the steepest-descent method. *True*
2. Conjugate directions are computed from gradients of the cost function. *True*
3. Conjugate directions are normal to each other. *False*
4. The conjugate direction at the kth point is orthogonal to the gradient of the cost function at the (k+l)th point when an exact step size is calculated. *True*
5. The conjugate direction at the kth point is orthogonal to the gradient of the cost function at the (k−1)th point. *False*

Note: In the solution of Exercises 10.67 - 10.76, the first iteration is the same as the Steepest Descent Method given in the solution of Exercises 10.52 – 10.61, respectively.

10.67

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1, x_2) = x_1^2 + 2x_2^2 - 4x_1 - 2x_1x_2;\ \text{starting point, } \mathbf{x} = [1,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [2x_1 - 4 - 2x_2, 4x_2 - 2x_1] = [-4, 2], \left\|\mathbf{c}^{(0)}\right\| = \sqrt{20} > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [4, -2]$
4. Calculate a step size α:
   $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1,1] + \alpha[4, -2] = [1 + 4\alpha, 1 - 2\alpha]$
   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 40\alpha^2 - 20\alpha - 3$
   $f'(\alpha) = 80\alpha - 20 = 0, \alpha_0 = 0.25$
   $f''(\alpha_0) = 80 > 0,$ So α_0 = 0.25 is a minimum point.
5. Update the design:
   $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [2, 0.5], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [2, 0.5], k = 1$
2. $\mathbf{c}^{(1)} = [-1, -2], \left\|\mathbf{c}^{(1)}\right\| = \sqrt{5} > \varepsilon$
3. $\beta_1 = \left(\left\|\mathbf{c}^{(1)}\right\| / \left\|\mathbf{c}^{(0)}\right\|\right)^2 = 0.25; \mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [1, 2] + 0.25[4, -2] = [2, 1.5]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [2 + 2\alpha, 0.5 + 1.5\alpha];$
   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 2.5\alpha^2 - 5\alpha - 5.5$
   $f'(\alpha) = 5\alpha - 5 = 0, \alpha_1 = 1$
5. $\mathbf{x}^{(2)} = [4, 2], k = k + 1$

10.68

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1, x_2) = 12.096x_1^2 + 21.504\,x_2^2 - 1.7321x_1 - x_2; \text{ starting point, } \mathbf{x} = [1,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [24.192x_1 - 1.7321, 43.008\,x_2 - 1] = [22.4599, 42.008], \left\|\mathbf{c}^{(0)}\right\| = 47.6 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-22.4599, -42.008]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1 - 22.4599\alpha, 1 - 42.008\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 12.096(1 - 22.4599\alpha)^2 + 21.504(1 - 42.008\alpha)^2 - 1.7321(1 - 22.4599\alpha)$
   $-(1 - 42.008\alpha)$

   $f\,'(\alpha) = -2269.1192 + 88098.6\alpha = 0, \alpha_0 = 0.025757$

   $f\,''(\alpha_0) = 88098.6 > 0$, So, α_0 = 0.025757 is a minimum point.
6. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.42151, -0.08198], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.42151, -0.08198], k = 1$
2. $\mathbf{c}^{(1)} = [8.465, -4.525], \left\|\mathbf{c}^{(1)}\right\| = 9.60 > \varepsilon$
3. $\beta_1 = \left(\left\|\mathbf{c}^{(1)}\right\| / \left\|\mathbf{c}^{(0)}\right\|\right)^2 = 0.0406; \mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [-8.465, 4.525] + 0.0406[-22.4599, -42.008]$
   $= [-9.3769, 2.8194]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.42151 - 9.3769\alpha, -0.08198 + 2.8194\alpha];$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 12.096(0.42151 - 9.3769\alpha)^2 + 21.504(-0.08198 + 2.8194\alpha)^2$
   $-1.7321(0.42151 - 9.3769\alpha) - (-0.08198 - 2.8194\alpha)$

   $f\,'(\alpha) = 2468.983\alpha - 86.498 = 0, \alpha_1 = 0.035$
5. $\mathbf{x}^{(2)} = [0.093319, 0.016699], k = k + 1$

10.69

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1, x_2) = 6.983x_1^2 + 12.415x_2^2 - x_1;\ \text{starting point, } \mathbf{x} = [2,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [2,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [13.966x_1 - 1, 24.83\text{x}_2] = [26.932, 24.83], \left\|\mathbf{c}^{(0)}\right\| = 36.6314 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-26.932, -24.83]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [2 - 26.932\alpha, 1 - 24.83\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 6.983(2 - 26.932\alpha)^2 + 12.415(1 - 24.83\alpha)^2 - (2 - 26.936\alpha)$

   $f\,'(\alpha) = 25441.4173\alpha - 1341.97325 = 0, \alpha_0 = 0.05275$

   $f\,''(\alpha_0) = 25441 > 0;$ So, α_0 = 0.05275 is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.579, -0.310], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.579, -0.310], k = 1$
2. $\mathbf{c}^{(1)} = [7.086, -7.697], \left\|\mathbf{c}^{(1)}\right\| = 10.4621 > \varepsilon$
3. $\beta_1 = \left(\left\|\mathbf{c}^{(1)}\right\| / \left\|\mathbf{c}^{(0)}\right\|\right)^2 = 0.08157; \mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [-7.086, 7.697] + 0.08157[-26.932, -24.83]$

   $= [-9.2828, 5.6716]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.579 - 9.2828\alpha, -0.310 + 5.6716\alpha];$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 6.983(0.579 - 9.2828\alpha)^2 + 12.415(-0.31 + 5.6716\alpha)^2$

   $-(0.579 - 9.2828\alpha)$

   $f\,'(\alpha) = 2002.16\alpha - 109.437 = 0, \alpha_1 = 0.05466$
5. $\mathbf{x}^{(2)} = [0.072, 0.0], k = k + 1$

10.70

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1, x_2) = 12.096x_1^2 + 21.504x_2^2 - x_2;\ \text{starting point, } \mathbf{x} = [1, 2]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1, 2], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [24.192x_1, 43.008\,\mathrm{x}_2 - 1] = [24.192, 85.016], \left\|\mathbf{c}^{(0)}\right\| = 88.391 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-24.192, -85.016]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1 - 24.192\alpha, 2 - 85.016\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 12.096(1 - 24.192\alpha)^2 + 21.504(2 - 85.016\alpha)^2 - (2 - 85.016\alpha)$

   $f\,'(\alpha) = 325008.23\alpha - 7812.973 = 0, \alpha_0 = 0.02404$

   $f\,''(\alpha_0) = 325008 > 0,$ So, α_0 = 0.02404 is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.418, -0.0437], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.418, -0.0437], k = 1$
2. $\mathbf{c}^{(1)} = [10.112, -2.879], \left\|\mathbf{c}^{(1)}\right\| = 10.514 > \varepsilon$
3. $\beta_1 = \left(\left\|\mathbf{c}^{(1)}\right\| / \left\|\mathbf{c}^{(0)}\right\|\right)^2 = 0.014148$

   $\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [-10.112, 2.879] + 0.014148[-24.192, -85.016] = [-9.2828, 5.6716]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.418 - 10.454\alpha, -0.0437 + 1.6761\alpha];$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 12.096(0.418 - 10.454\alpha)^2 + 21.504(-0.0437 + 1.6761\alpha)^2$
   $-(-0.0437 + 1.6761\alpha)$

   $f\,'(\alpha) = 2764.67\alpha - 110.54 = 0, \alpha_1 = 0.04$
5. $\mathbf{x}^{(2)} = [0.0, 0.0233], k = k + 1$

10.71

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1,x_2) = 25x_1^2 + 20x_2^2 - 2x_1 - x_2; \text{ starting point, } \mathbf{x} = [3,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [3,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [50x_1 - 2, 40\,\mathrm{x}_2 - 1] = [148, 39], \left\|\mathbf{c}^{(0)}\right\| = 153.05 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-148, -39]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [3 - 148\alpha, 1 - 39\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 25(3 - 148\alpha)^2 + 20(1 - 39\alpha)^2 - 2(3 - 148\alpha) - (1 - 39\alpha)$

   $f\,'(\alpha) = 1156040\alpha - 23425 = 0, \alpha_0 = 0.02026$

   $f\,''(\alpha_0) = 1156040 > 0$; So, α_0 = 0.02026 is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.001055, 0.2097], k = k + 1$

Iteration 2:

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.001055, 0.2097], k = 1$
2. $\mathbf{c}^{(1)} = [-1.947, 7.388], \left\|\mathbf{c}^{(1)}\right\| = 7.6402 > \varepsilon$
3. $\beta_1 = \left(\left\|\mathbf{c}^{(1)}\right\| / \left\|\mathbf{c}^{(0)}\right\|\right)^2 = 0.0025$

   $\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [1.947, -7.388] + 0.0025[-148, -39] = [1.578, -7.485]$
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.001055 + 1.578\alpha, 0.2097 - 7.485\alpha]$;

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 25(0.001055 + 1.578\alpha)^2 + 20(0.2097 - 7.485\alpha)^2$

   $-2(0.001055 + 1.578\alpha) - (0.2097 - 7.485\alpha)$

   $f\,'(\alpha) = 2365.51\alpha - 58.372 = 0, \alpha_1 = 0.02468$
5. $\mathbf{x}^{(2)} = [0.040, 0.025], k = k + 1$

10.72

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3; \text{ starting point, } \mathbf{x} = [1,1,1]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1,1,1], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [2x_1 + 2x_2, 4x_2 + 2x_1 + 2x_3, 4x_3 + 2x_2] = [4,8,6], \left\|\mathbf{c}^{(0)}\right\| = 10.77 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-4,-8,-6]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1-4\alpha, 1-8\alpha, 1-6\alpha]$
   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = (1-4\alpha)^2 + 2(1-8\alpha)^2 + 2(1-6\alpha)^2 + 2(1-4\alpha)(1-8\alpha) + 2(1-8\alpha)(1-6\alpha)$
   $f'(\alpha) = 752\alpha - 116 = 0, \alpha_0 = 0.154255$
   $f''(\alpha_0) = 752 > 0;$ So, $\alpha_0 = 0.154255$ is a minimum point.
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [0.383, -0.234, 0.0745], k = k+1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [0.383, -0.234, 0.0745], k = 1$
2. $\mathbf{c}^{(1)} = [0.298, -0.021, -0.17], \left\|\mathbf{c}^{(1)}\right\| = 0.3437 > \varepsilon$
3. $\beta_1 = \left(\left\|\mathbf{c}^{(1)}\right\| / \left\|\mathbf{c}^{(0)}\right\|\right)^2 = 0.001$
   $\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [-0.298, 0.021, 0.17] + 0.001[-4,-8,-6] = [-0.302, 0.013, 0.164]$
4. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [0.383 - 0.302\alpha, -0.234 + 0.013\alpha, 0.0745 + 0.164\alpha];$
   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = (0.383 - 0.302\alpha)^2 + 2(-0.234 + 0.013\alpha)^2 + 2(0.0745 + 0.164\alpha)^2$
   $+2(-0.234 + 0.013\alpha)(0.4575 - 0.138\alpha)$
   $f'(\alpha) = 0.28349\alpha - 0.118149 = 0, \alpha_1 = 0.416763$
5. $\mathbf{x}^{(2)} = [0.257, -0.229, 0.143], k = k+1$

10.73

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1, x_2) = 8x_1^2 + 8x_2^2 - 80\sqrt{x_1^2 + x_2^2 - 20x_2 + 100} + 80\sqrt{x_1^2 + x_2^2 + 20x_2 + 100} - 5\,\mathrm{x}_1 - 5\,\mathrm{x}_2;$$

starting point, $\mathbf{x} = [4, 6]$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [4, 6], k = 0, \varepsilon = 0.001$
2. $\mathbf{c}^{(0)} = [-16.9714, 69.9571], \left\|\mathbf{c}^{(0)}\right\| = 71.9863 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [16.9714, -69.9571]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [4 + 16.9714\alpha, 6 - 69.9571\alpha]$

$$f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 8(4 + 16.9714\alpha)^2 + 8(6 - 69.9571\alpha)^2 - 5(10 - 52.9857\alpha)$$
$$-80\sqrt{(4 + 16.9714\alpha)^2 + (6 - 69.9571\alpha)^2 - 20(6 - 69.9571\alpha) + 100}$$
$$-80\sqrt{(4 + 16.9714\alpha)^2 + (6 - 69.9571\alpha)^2 + 20(6 - 69.9571\alpha) + 100}$$

(Using the Golden Section Method in Appendix B to solve for α_0); $\alpha_0 = 0.07885$

5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [5.3382, 0.4839], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [5.3382, 0.4839], k = 1$
2. $\mathbf{c}^{(1)} = [4.9720, 1.2237], \left\|\mathbf{c}^{(1)}\right\| = 5.124 > \varepsilon$
3. $\beta_1 = \left(\left\|\mathbf{c}^{(1)}\right\| / \left\|\mathbf{c}^{(0)}\right\|\right)^2 = 0.005967$

$$\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [-4.9720, -1.2237] + 0.005967[16.9714, -69.9571] = [-4.8707, -1.6411]$$

5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [5.3382 - 4.8707\alpha, 0.4839 - 1.6411\alpha];$

$$f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 8(5.3382 - 4.8707\alpha)^2 + 8(0.4839 - 1.6411\alpha)^2 - 5(5.8221 - 6.5118\alpha)$$
$$-80\sqrt{(5.3382 - 4.8707\alpha)^2 + (0.4839 - 1.6411\alpha)^2 - 20(0.4839 - 1.6411\alpha) + 100}$$
$$+80\sqrt{(5.3382 - 4.8707\alpha)^2 + (0.4839 - 1.6411\alpha)^2 + 20(0.4839 - 1.6411\alpha) + 100}$$

$\alpha_1 = 0.0620029$ (Golden Section Search)

5. $\mathbf{x}^{(2)} = [5.0362, 0.3821], k = k + 1$

10.74

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1,x_2)=9x_1^2+9x_2^2-100\sqrt{x_1^2+x_2^2-20x_2+100}-64\sqrt{x_1^2+x_2^2+16x_2+64}-5\,\mathrm{x}_1-41\,\mathrm{x}_2;$$

starting point, $\mathbf{x}=[5,2]$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)}=[5,2], k=0, \varepsilon=0.001$
2. $\mathbf{c}^{(0)}=[3.378436, 22.55649], \left\|\mathbf{c}^{(0)}\right\|=22.8>\varepsilon$
3. $\mathbf{d}^{(0)}=-\mathbf{c}^{(0)}=[-3.378436,-22.55649]$
4. $\mathbf{x}^{(1)}=\mathbf{x}^{(0)}+\alpha\mathbf{d}^{(0)}=[5-3.378436\alpha, 2-22.55649\alpha]$

$$f(\alpha)=f(\mathbf{x}^{(0)}+\alpha\mathbf{d}^{(0)})=9(5-3.378436\alpha)^2+9(2-22.55649\alpha)^2-5(5-3.378436\alpha)$$
$$-41(2-22.55649\alpha)-100\sqrt{(5-3.378436\alpha)^2+(2-22.55649\alpha)^2-20(2-22.55649\alpha)+100}$$
$$-64\sqrt{(5-3.378436\alpha)^2+(2-22.55649\alpha)^2+16(2-22.55649\alpha)+64}$$

(Using the Golden Section Method in Appendix B to solve for α_0); $\alpha_0=0.07327$

5. $\mathbf{x}^{(1)}=\mathbf{x}^{(0)}+\alpha_0\mathbf{d}^{(0)}=[4.75246, 0.34729], k=k+1$

Iteration 2:

1. $\mathbf{x}^{(1)}=[4.75246, 0.34729], k=1$
2. $\mathbf{c}^{(1)}=[4.7079,-0.6505], \left\|\mathbf{c}^{(1)}\right\|=4.75>\varepsilon$
3. $\beta_1=\left(\left\|\mathbf{c}^{(1)}\right\|/\left\|\mathbf{c}^{(0)}\right\|\right)^2=0.043403$

$$\mathbf{d}^{(1)}=-\mathbf{c}^{(1)}+\beta_1\mathbf{d}^{(0)}=[-4.7079, 0.6505]+0.043403[-3.378436,-22.55649]$$
$$=[-4.8545,-0.3285]$$

4. $\mathbf{x}^{(2)}=\mathbf{x}^{(1)}+\alpha\mathbf{d}^{(1)}=[4.75246-4.8545\alpha, 0.34729-0.3285\alpha];$

$$f(\alpha)=f(\mathbf{x}^{(1)}+\alpha\mathbf{d}^{(1)})=9(4.75246-4.8545\alpha)^2+9(0.34729-0.3285\alpha)^2$$
$$-5(4.75246-4.8545\alpha)-41(0.34729-0.3285\alpha)$$
$$-100\sqrt{(4.75246-4.8545\alpha)^2+(0.34729-0.3285\alpha)^2-20(0.34729-0.3285\alpha)+100}$$
$$-64\sqrt{(4.75246-4.8545\alpha)^2+(0.34729-0.3285\alpha)^2+16(0.34729-0.3285\alpha)+64}$$

$\alpha_1=0.208686$ (Golden Section Search)

5. $\mathbf{x}^{(2)}=[3.7394, 0.2787], k=k+1$

10.75

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2; \text{ starting point, } \mathbf{x} = [5, 2]$$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [5, 2], k = 0, \varepsilon = 0.00001$
2. $\mathbf{c}^{(0)} = [x_1\left(400x_1^2 + 2\right) - \left(400x_1x_2\right) - 2, 200x_2 - 200x_1^2] = [46008, -4600], \left\|\mathbf{c}^{(0)}\right\| = 46237 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [-46008, 4600]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [5 - 46008\alpha, 2 + 4600\alpha]$

   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = 100\left((2 + 4600\alpha) - (5 - 46008\alpha)^2\right)^2 + \left(1 - (5 - 46008\alpha)\right)^2$

   (Using the Golden Section Method in Appendix B to solve for α_0); $\alpha_0 = 0.00014$
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [-1.44, 2.644], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [-1.44, 2.644], k = 1$
2. $\mathbf{c}^{(1)} = [323.67, 114.08], \left\|\mathbf{c}^{(1)}\right\| = 343.19 > \varepsilon$
3. $\beta_1 = \left(\left\|\mathbf{c}^{(1)}\right\| / \left\|\mathbf{c}^{(0)}\right\|\right)^2 = 0.0000551$

   $\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [-323.67, -114.08] + 0.0000551[-46008, 4600] = [-326.205, -113.8265]$
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [-1.44 - 326.205\alpha, 2.644 - 113.8265\alpha];$

   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = 100\left((2.644 - 113.8265\alpha) - (-1.44 - 326.205\alpha)^2\right)^2$

   $+\left(1 - (-1.44 - 326.205\alpha)\right)^2$

   $\alpha_1 = 0.000329014$ (Golden Section Search)
5. $\mathbf{x}^{(2)} = [-1.5473, 2.6065], k = k + 1$

10.76

For the following function, complete two iterations of the conjugate gradient method starting from the given design point.

$$f(x_1, x_2, x_3, x_4) = (x_1 - 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4;$$

starting point, $\mathbf{x} = [1, 2, 3, 4]$

Solution:

Iteration 1:

1. $\mathbf{x}^{(0)} = [1, 2, 3, 4], k = 0, \varepsilon = 0.00001$
2. $\mathbf{c}^{(0)} = [2(x_1 - 10x_2) + 40(x_1 - x_4)^3, -20(x_1 - 10x_2) + 4(x_2 - 2x_3)^3, 10(x_3 - x_4) - 8(x_2 - 2x_3)^3,$
   $-10(x_3 - x_4) - 40(x_1 - x_4)^3] = [-1118, 124, 502, 1090], \|\mathbf{c}^{(0)}\| = 1645 > \varepsilon$
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = [1118, -124, -502, -1090]$
4. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = [1 + 1118\alpha, 2 - 124\alpha, 3 - 502\alpha, 4 - 1090\alpha]$
   $f(\alpha) = f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = (2358\alpha - 19)^2 + 5(588\alpha - 1)^2 + (880\alpha - 4)^4 + 10(2208\alpha - 3)^4$
   (Using the Golden Section Method in Appendix B to solve for α_0); $\alpha_0 = 0.001854$
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = [3.0728, 1.7701, 2.06929, 1.97914], k = k + 1$

Iteration 2:

1. $\mathbf{x}^{(1)} = [3.0728, 1.7701, 2.06929, 1.97914], k = 1$
2. $\mathbf{c}^{(1)} = [23.068, 239.418, 107.193, -53.226], \|\mathbf{c}^{(1)}\| = 268.66 > \varepsilon$
3. $\beta_1 = \left(\|\mathbf{c}^{(1)}\| / \|\mathbf{c}^{(0)}\|\right)^2 = 0.0266731$
   $\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = [-23.068, -239.418, -107.193, 53.226]$
   $+0.0266731[1118, -124, -502, -1090] = [6.7524, -242.7255, -120.5829, 24.1524]$
6. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)} = [3.0728 + 6.7524\alpha, 1.7701 - 242.7255\alpha,$
   $2.06929 - 120.5829\alpha, 1.97914 + 24.1524\alpha]$
   $f(\alpha) = f(\mathbf{x}^{(1)} + \alpha\mathbf{d}^{(1)}) = (-14.6282 + 2434.0074\alpha)^2 + 5(0.09015 - 144.7353\alpha)^2$
   $+(-2.36848 - 1.5597\alpha)^4 + 10(1.09366 - 17.4\alpha)^4$
   $\alpha_1 = 0.0059028$ (Golden Section Search)
5. $\mathbf{x}^{(2)} = [3.113, 0.3374, 1.3574, 2.1214], k = k + 1$

10.77

Write a computer program to implement the conjugate gradient method (or, modify the steepest-descent program given in Appendix B). Solve Exercises 10.52 to 10.61 using your program.

Solutions:

The solutions were obtained using the following program, written in C++.

```
#include <iostream>
#include <math.h>

using namespace std;

void goldenIntervalSearch(float x[], float xn[], float d[], float *a, float delta,
    float epsilon, float *f, int *nCount, int ndv);
float funct(float xVector[], int *nC);
void grad(float xVec[], float grad[]);
void scale(float arrOne[], float arrTwo[], int scalar, int n);
float tNorm(float vec[], int n);
void update(float x[], float xwrk[], float d[], float a, int n);
void printStuff(int currCount, float x[], float alph, float grad[], float f, int
    ndv);

int main()
{
    float delta = 0.05;       // * modify this value for the required delta for
    line search
    float epsilon = 0.001;    // * modify this value for the required line search
    accuracy
    float epsilon2 = 0.001;   // * modify this value for steepest-descent method
    stopping criterion
    int ndv = 2;          // * modify this value for number of design variables
    int nCount = 0;       //   tracks number of function evaluations
    int noc = 1000;       // * modify this value for number of cycles of the method
    int nocActual = 0;    //   tracks number of cycles of the method

    float f = 0;           //   holds the value of the function at the minimum
    float x[2] = {1, 1};   // * modify this array to hold design variable vector
    float d[2] = {0, 0};        // * modify this array to hold the direction vector
    float c[2] = {0, 0};        //   holds the gradient vector
    float wrk[2] = {0, 0};      //   vector to hold temporary values
    float alpha = 0;            //   holds the value of alpha at the minimum
    float cmag1 = 0;
    float cmag2 = 0;
    float beta = 0;

    float *alphaptr = &alpha;   //  pointer to alpha
    float *fptr = &f;           //  pointer to f
    int *nptr = &nCount;        //  pointer to nCount

    f = funct(x, nptr);
    grad(x, c);
    for(int z = 0; z < ndv; z++)
    {
        d[z] = -1*c[z];
    }

    for(int k = 1; k <= noc; k++)
    {
        cmag1 = tNorm(c, ndv);
        printStuff(k, x, alpha, c, f, ndv);

        if(cmag1 <= epsilon2)
        {
            k = noc+1;
            for(int l = 0; l < ndv; l++)
            {
                cout<<"x["<<l<<"] = "<<x[l]<<"\n";
            }
        }
        else if(cmag1 > epsilon2)
        {
            goldenIntervalSearch(x, wrk, d, alphaptr, delta, epsilon, fptr, nptr,
    ndv);
            update(x, x, d, alpha, ndv);
            nocActual = k;
            f = funct(x, nptr);
            grad(x, c);
            cmag2 = tNorm(c, ndv);
            beta = pow(cmag2/cmag1, 2);
            scale(c, d, beta, ndv);
        }
    }

    if(nocActual >= noc)
    {
        cout<<"Limit on number of cycles has exceeded maximum."<<"\n";
        for(int l = 0; l < ndv; l++)
        {
            cout<<"x["<<l<<"] = "<<x[l]<<"\n";
        }
    }

    cout <<"f* = "<<f<<"\n";                //outputs optimum cost function value
    cout <<"NFE: "<<nCount<<"\n";     //outputs the number of function evaluations
    cout<<"NIT: "<<nocActual;

    return 0;
}

/* this function implements golden interval search
    a = optimal value of alpha on return
    d = initial step length
    e = convergence parameter
    f = optimum value of the function on return
    n = number of function evaluations on return
**/
void goldenIntervalSearch(float x[], float xn[], float d[], float *a, float delta,
    float epsilon, float *f, int *nCount, int ndv)
{
    int q = 1;                          //delta multiplier
    float gr = 0.5*pow(5,0.5)+0.5;      //holds value of golden section multiplier
    float delta1 = delta;               //holds optimal value of delta, initially
    set to passed value
    float al = 0;                       //lower bound of alpha
    float aa = delta1;                  //holds value of mid-point alpha a,
    initially set to delta1
    float ab = 0;                         //holds value of mid-point alpha b
    float au = aa+(pow(gr, q)*delta1);    //holds value of upper bound of alpha
    float fl = 0;
    float fa = 0;                         //holds function value of mid-point alpha
    float fb = 0;                       //holds function value at mid-point alpha b
    float fu = 0;                       //holds function value of upper alpha bound

    update(x, xn, d, al, ndv);          //call function update() to update the
    value of xn
    fl = funct(xn, nCount);             //call function funct() to return the value
    at a given value of alpha
    update(x, xn, d, aa, ndv);
    fa = funct(xn, nCount);
    update(x, xn, d, au, ndv);
    fu = funct(xn, nCount);

    //establish initial interval of uncertainty
    while(fa > fu)
    {
        q = q+1;
        al = aa;
        aa = au;
        au = au+(pow((float)1.618, q)*delta1);
        fl = fa;
        fa = fu;
        update(x, xn, d, au, ndv);
        fu = funct(xn, nCount);
    }

    ab = al+((au-al)/gr);
    update(x, xn, d, ab, ndv);
    fb = funct(xn, nCount);

    //refine the interval of uncertainty further
    while((au-al) >= epsilon)
    {
        int caseInPoint = 0;

        if(fa < fb)
            caseInPoint = 1;
        else if(fa > fb)
            caseInPoint = 2;
        else if(fa == fb)
            caseInPoint = 3;

        switch(caseInPoint)
        {
            case 1:
                au = ab;
                ab = aa;
                fb = fa;
                fu = fb;
                aa = al+((1-1/gr)*(au-al));
                update(x, xn, d, aa, ndv);
                fa = funct(xn, nCount);
                break;
            case 2:
                al = aa;
                aa = ab;
                fl = fa;
                fa = fb;
                ab = al+((au-al)/gr);
                update(x, xn, d, ab, ndv);
                fb = funct(xn, nCount);
                break;
            case 3:
                al = aa;
                au = ab;
                fl = fa;
                fu = fb;
                aa = al+((1-1/gr)*(au-al));
                update(x, xn, d, aa, ndv);
                fa = funct(xn, nCount);
                ab = al+((au-al)/gr);
                update(x, xn, d, ab, ndv);
                fb = funct(xn, nCount);
                break;
            default:
                break;
        }

    }

    *a = (au+al)/(float)2;      //send minimum alpha value to pointer address
}

/* calculates the function value
    a = value of alpha, input
    fVal = function value on return
    nC = number of calls for function evaluation
**/
float funct(float xVector[], int *nC)
{
    float fVal = 0;
    float x1 = xVector[0];
    float x2 = xVector[1];
    *nC = *nC + 1;              //send function evaluation count to pointer address
    fVal = pow(x1, 2)+2*pow(x2, 2)-4*x1-2*x1*x2;     //enter function of one
    variable here

    return fVal;
}

void grad(float xVec[], float grad[])
{
    float x1 = xVec[0];
    float x2 = xVec[1];
    grad[0] = 2*x1-4-2*x2;
    grad[1] = 4*x2-2*x1;
```

```
}

void scale(float arrOne[], float arrTwo[], int scalar, int n)
{
    for(int i = 0; i < n; i++)
    {
        arrTwo[i] = -1*arrOne[i]+scalar*arrTwo[i];
    }
}

float tNorm(float vec[], int n)
{
    float sum = 0;
    float tNorm = 0;

    for(int i = 0; i < n; i++)
    {
        sum = sum+pow(vec[i], 2);
    }

    tNorm = pow(sum, 0.5);

    return tNorm;
}


void update(float x[], float xwrk[], float d[], float a, int n)
{
    for(int i = 0; i < n; i++)
    {
        xwrk[i] = x[i]+(a*d[i]);
    }
}

void printStuff(int currCount, float x[], float alph, float grad[], float f, int
    ndv)
{
    cout<<"Iteration number: "<<currCount<<" Function value: "<<f<<" Alpha value:
    "<<alph<<"\n";
    cout<<"Norm of gradient vector: "<<tNorm(grad, ndv)<<"\n";
}
```

10.52: **x*** = (3.99891, 1.99935), *f* * = − 8.0, NIT = 4, NFE = 80;
10.53: **x*** = (0.0716066, 0.0232642), *f* * = − 0.0736332, NIT = 8, NFE = 125;
10.54: **x*** = (0.071618, -0.0000132951), *f* * = − 0.0358012, NIT = 9, NFE = 141;
10.55: **x*** = (0.00000998224, 0.0232725), *f* * = − 0.0116257, NIT = 6, NFE = 96;
10.56: **x*** = (0.0399866, 0.025), *f* * = − 0.0525, NIT = 3, NFE = 49;
10.57: **x*** = (0.000494682, -0.000290992, 0.000167161), *f* * = 0.0, NIT = 18, NFE = 388;
10.58: **x*** = (4.14491, 0.361561), *f* * = −1616.18, NIT = 19, NFE = 265;
10.59: **x*** = (3.73312, 0.341153), *f* * = −1526.56, NIT = 16, NFE = 227;
10.60: **x*** = (0.999062, 0.998119), *f* * = 0.000000882718, NIT = 14, NFE = 225;
10.61: **x*** = (0.0488862, 0.00489234, 0.0264347, 0.0264981), *f* * = 0.00000783204, NIT = 80, NFE = 1312.

Note: In the solution of Exercises 10.78 - 10.87(Exercises 10.52 – 10.61, respectively), use Excel-Solver.

Solution:

An example is shown for the solution of Exercise 10.52 (Exercise 10.78)

x1	1
x2	1
function:	-3

x1	4
x2	2
function:	-8

The table on the left shows the variable values and the function value before the solver is run, and the table on the right shows the values after the solver is run.

The Excel Answer Report is also shown below

Microsoft Excel 15.0 Answer Report

Worksheet: [Book2]Sheet1

Report Created: 7/30/2014 4:11:04 PM

Result: Solver found a solution. All Constraints and optimality conditions are satisfied.

Solver Engine

Engine: GRG Nonlinear

Solution Time: 0.016 Seconds.

Iterations: 3 Subproblems: 0

Solver Options

Max Time Unlimited, Iterations Unlimited, Precision 0.000001

Convergence 0.0001, Population Size 100, Random Seed 0, Derivatives Central

Max Subproblems Unlimited, Max Integer Sols Unlimited, Integer Tolerance 1%

Objective Cell (Min)

Cell	Name	Original Value	Final Value
C6	function:	-3	-8

Variable Cells

Cell	Name	Original Value	Final Value	Integer
C3	x1	1	4	Contin
C4	x2	1	2	Contin

Constraints

NONE