Mathematics 4670: Assignment #3

Due on Friday, September 22, 2016

Dr.Glunt

Michael Timbes

Contents

Problem 1

In this problem, I have to find the vector version of the planar equation $x + y + 2z = 10$. Which results in $\vec{n} < 1, 1, 2 >$ and given the point, $P(1, 2, 1)$ I am asked to find the perpendicular distance. The following steps are how I approached the question and these steps serve as the basis for the FORTRAN program to follow.

1. First, find the normal vector and its magnitude given the planar equation.

   The normal vector $\vec{n}(1, 1, 2)$ is given by the planar equation's coefficients.

   The magnitude is found by the following equation.

$$\parallel \vec{n} \parallel = \sqrt{x^2 + y^2 + z^2} = \sqrt{6} \tag{1}$$

2. Second, find a point on the plane(which I just set y and z equal to zero) and then make a vector $\vec{PQ}$ that goes through the points.

   $P\,(1, 2, 1)$ and $Q\,(1, 0, 0)$ so that $\vec{PQ} = \langle 0, 2, 1\rangle$

3. The third step is to find the absolute value of the inner product of $\vec{PQ}$ and the normal vector of the plane. The equation for that is: $abs\left(\vec{PQ} \cdot \vec{n}\right)$

4. Since the perpendicular distance between a point and a plane is the projection of a point on the plane to a point near the plane on to the normal of the plane in question, I will use that formula.

$$proj_n \vec{PQ} = \frac{abs\left(\vec{PQ} \cdot \vec{n}\right)}{\parallel \vec{n} \parallel} = \frac{4}{\sqrt{6}} \tag{2}$$

In describing the reasoning behind the program I will begin with the variables that I decided to use. First, most of the variables are double precision. I have the arrays that are dynamic which I don't really need- I could have them fixed but I always aim to be as modular as I can when I write. There are two vectors: 'V' which is the planar vector(also the normal), 'PQ' which is the vector created joining 'P' and 'Q', and two points 'P' and 'Q'. The V_x, y, z variables are the components of the 'V' vector and likewise with the 'P' components for the point. The 'DIST' is for the distance calculation,'D' for dimension,'I' just for an iterator, and the others are self explanatory.
Below the main are the three subroutines: FINDPQ,DOTPROD,and MAGV. The FindPQ is passed the points P and Q, PQ is passed to update its value,and D is passed to ensure no outside bounds errors occur. The PQ is found by subtracting each component of P from each component of Q to get a line. DOTPROD is passed PQ,V,DOTRESULT,and D, its function is to update DOTRESULT to reflect the absolute value of the inner product of PQ and V; the inner product is calculated by adding the product of the components of the two vectors. MAGV is passed V(for the normal),MAGNITUDE (to store the result), and D. The purpose of MAGV is to find the magnitude of the normal vector by add the square of each components and then taking the square root of the result.

Problem 2

Listing 1: Code for perpendicular distance

```
PROGRAM VECTORDIS
IMPLICIT NONE

DOUBLE PRECISION, DIMENSION(:), ALLOCATABLE :: V,P,Q,PQ
DOUBLE PRECISION :: V_X,V_Y,V_Z,DOTRESULT,MAGNITUDE=0,DIST
DOUBLE PRECISION :: P_X,P_Y,P_Z
!INTEGERS FOR DO INDEXING and TO STORE THE DIMENSION
!(D)- SOMETHING THAT CAN BE ASSIGNED FOR MORE MODULAR PROGRAM
INTEGER :: I,D=3

!**************************************************************************!
!Explicit Interface- Requirement for passing allocatable argument as safe gaurd!
INTERFACE
SUBROUTINE FIND_PQ (P,Q,PQ,D)
!SET UP VALUES FOR THE RETURN ARRAY- NOT A REF TO OUTSIDE FUNCTION- INTENT NOT NEEDED
DOUBLE PRECISION, DIMENSION(:), ALLOCATABLE :: P,Q,PQ
INTEGER::I,D
END SUBROUTINE FIND_PQ
END INTERFACE
!**************************************************************************!

!**************************************************************************!
!Explicit Interface- Requirement for passing allocatable argument as safe gaurd!
INTERFACE
SUBROUTINE DOTPROD (PQ,V,DOTRESULT,D)
IMPLICIT NONE
!SET UP VALUES FOR THE RETURN ARRAY
DOUBLE PRECISION, DIMENSION(:), ALLOCATABLE :: V,PQ
DOUBLE PRECISION :: DOTRESULT
INTEGER::I,D
END SUBROUTINE DOTPROD
```

```
END INTERFACE
!**************************************************************************!

!**************************************************************************!
!Explicit Interface- Requirement for passing allocatable argument as safe gaurd!
INTERFACE
SUBROUTINE MAGV (V,MAGNITUDE,D)
IMPLICIT NONE
!SET UP VALUES FOR THE RETURN ARRAY
DOUBLE PRECISION, DIMENSION(:), ALLOCATABLE :: V
DOUBLE PRECISION :: MAGNITUDE
INTEGER::I,D
END SUBROUTINE MAGV
END INTERFACE
!**************************************************************************!



!PRINT*,"ENTER THE DIMENSIONS OF THE ARRAYS: "
!READ*,D

!BEGIN INPUT FOR PLANE, SPECIFY 3-D,READ VALS, INPUT TO V
PRINT*,"INPUT PLANE VECTOR:" ,D ,"-D"
READ*,V_X,V_Y,V_Z
 allocate(V(D))
V(1)=V_X
V(2)=V_Y
V(3)=V_Z

!POINT Q ON PLANE X=1,Y=0,Z=0
ALLOCATE(Q(D))
Q(1) = V(1)
Q(2) = 0
Q(3) = 0

!BEGIN INPUT FOR THE POINT P(X,Y,Z)
PRINT*,"INPUT THE POINT" ,D ,"-D"
READ*,P_X,P_Y,P_Z
 allocate(P(D))
P(1)=P_X
P(2)=P_Y
P(3)=P_Z

!ALLOCATE FOR THE RETURN ARRAY PQ
ALLOCATE(PQ(D))
!IF STATEMENT USED TO ENSURE THE ARRAYS ARE ALLOCATED
IF(ALLOCATED(PQ).AND.ALLOCATED(P).AND.ALLOCATED(Q)) THEN
CALL FIND_PQ(P,Q,PQ,D)
ENDIF

!ALLOCATE FOR THE RETURN ARRAY DOTRESULT
!IF STATEMENT USED TO ENSURE THE ARRAYS ARE ALLOCATED
IF(ALLOCATED(PQ).AND.ALLOCATED(V)) THEN
CALL DOTPROD (PQ,V,DOTRESULT,D)
```

```
ENDIF

!ALLOCATE FOR THE RETURN ARRAY DOTRESULT
!IF STATEMENT USED TO ENSURE THE ARRAYS ARE ALLOCATED
IF(ALLOCATED(V)) THEN
CALL MAGV (V,MAGNITUDE,D)
ENDIF

!DEFINE DISTANCE
DIST = DOTRESULT/MAGNITUDE

!PRINT THE ELEMENTS
PRINT*,"PLANE VECTOR:"
PRINT*,V(1),V(2),V(3)
PRINT*,"POINT:"
PRINT*,P(1),P(2),P(3)
PRINT*,"VECTOR PQ:"
PRINT*,PQ(1),PQ(2),PQ(3)
PRINT*,"DOTPRODUCT:",DOTRESULT
PRINT*,"MAGNITUDE:",MAGNITUDE
PRINT*,"DISTANCE:",DIST
!DEALLOCATION PROCESS
DEALLOCATE(V,P,Q,PQ)
END Program VECTORDIS


!FIND THE LINE PQ
SUBROUTINE FIND_PQ (P,Q,PQ,D)
IMPLICIT NONE
!SET UP VALUES FOR THE RETURN ARRAY
DOUBLE PRECISION, DIMENSION(:), ALLOCATABLE :: P,Q,PQ
INTEGER::I,D

!IF STATEMENT USED TO ENSURE THE ARRAYS ARE ALLOCATED
     IF(ALLOCATED(PQ).AND.ALLOCATED(P).AND.ALLOCATED(Q)) THEN
          DO I=1,D
                PQ(I) = P(I)-Q(I)
          END DO
     ENDIF
END SUBROUTINE




!FIND THE MAGNITUDE OF NORMAL VECTOR (AKA V)
SUBROUTINE MAGV (V,MAGNITUDE,D)
IMPLICIT NONE
DOUBLE PRECISION, DIMENSION(:), ALLOCATABLE :: V
DOUBLE PRECISION :: MAGNITUDE
INTEGER::I,D

!IF STATEMENT USED TO ENSURE THE ARRAYS ARE ALLOCATED
     IF(ALLOCATED(V)) THEN
```

```
          DO I=1,D
               MAGNITUDE = MAGNITUDE + (V(I)**2)
          END DO
MAGNITUDE = SQRT(MAGNITUDE)
     ENDIF
END SUBROUTINE




!DOT PRODUCT FUNCTION
SUBROUTINE DOTPROD (PQ,V,DOTRESULT,D)
IMPLICIT NONE
!SET UP VALUES FOR THE RETURN ARRAY
DOUBLE PRECISION, DIMENSION(:), ALLOCATABLE :: V,PQ
DOUBLE PRECISION :: DOTRESULT
INTEGER::I,D

!IF STATEMENT USED TO ENSURE THE ARRAYS ARE ALLOCATED
     IF (ALLOCATED(PQ).AND.ALLOCATED(V)) THEN
          DO I=1,D
               DOTRESULT = DOTRESULT + (PQ(I)*V(I))
          END DO
DOTRESULT = ABS(DOTRESULT)
     ENDIF
END SUBROUTINE
```

When comparing the hand calculations with the FORTRAN code, both answers were the same. Below are a few more examples with various ranges for the points and the plane.

Problem 3

```
TEST 1:

INPUT PLANE VECTOR:            3 -D
2 6 4
 INPUT THE POINT           3 -D
8 2 3
 PLANE VECTOR:
   2.0000000000000000        6.0000000000000000        4.0000000000000000
 POINT:
   8.0000000000000000        2.0000000000000000        3.0000000000000000
 VECTOR PQ:
   6.0000000000000000        2.0000000000000000        3.0000000000000000
 DOTPRODUCT:   36.000000000000000
 MAGNITUDE:   7.4833147735478827
 DISTANCE:   4.8107023544236389

TEST 2:

INPUT PLANE VECTOR:            3 -D
12 45 100
 INPUT THE POINT           3 -D
16 55 23
 PLANE VECTOR:
   12.000000000000000        45.000000000000000        100.00000000000000
 POINT:
   16.000000000000000        55.000000000000000        23.000000000000000
 VECTOR PQ:
   4.0000000000000000        55.000000000000000        23.000000000000000
 DOTPRODUCT:   4823.0000000000000
 MAGNITUDE:   110.31319050775387
 DISTANCE:   43.720972784854709

TEST 3:
INPUT PLANE VECTOR:            3 -D
20 48 123
 INPUT THE POINT           3 -D
200 465 11
 PLANE VECTOR:
   20.000000000000000        48.000000000000000        123.00000000000000
 POINT:
   200.00000000000000        465.00000000000000        11.000000000000000
 VECTOR PQ:
   180.00000000000000        465.00000000000000        11.000000000000000
 DOTPRODUCT:   27273.000000000000
```

```
 MAGNITUDE:   133.54025610279470
 DISTANCE:   204.23055036682109

TEST 4:
INPUT PLANE VECTOR:           3 -D
-1 -22 -6
 INPUT THE POINT           3 -D
1 23 24
 PLANE VECTOR:
  -1.0000000000000000      -22.000000000000000      -6.0000000000000000
 POINT:
   1.0000000000000000       23.000000000000000       24.000000000000000
 VECTOR PQ:
   2.0000000000000000       23.000000000000000       24.000000000000000
 DOTPRODUCT:   652.00000000000000
 MAGNITUDE:   22.825424421026653
 DISTANCE:   28.564638622858695

TEST 5:
INPUT PLANE VECTOR:           3 -D
2.555 12.465 .08
 INPUT THE POINT           3 -D
1 23.5 46.211
 PLANE VECTOR:
   2.5550000000000002       12.465000000000000       8.0000000000000002E-002
 POINT:
   1.0000000000000000       23.500000000000000       46.210999999999999
 VECTOR PQ:
  -1.5550000000000002       23.500000000000000       46.210999999999999
 DOTPRODUCT:   292.65135500000002
 MAGNITUDE:   12.724411577750857
 DISTANCE:   22.999205362997895
```