1. Program code

```
module form
        double precision m,c,k,xi,wn,wd,Tn,dt
        double precision,allocatable :: dis(:),vel(:),acc(:),p(:)
        double precision,parameter :: pi=3.141592d0
end module form

program hw_ch5
        use form
        implicit none
        integer choose,indat,outdat,i
        integer n
        print*, "Input natural period!"
        read*, Tn

        n=2048
        xi=0.02d0
        m=1.d0
        dt=0.02d0
        wn=2.d0*pi/Tn
        wd=wn*dsqrt(1.d0-xi**2.d0)
        c=2.d0*m*wn*xi
        k=m*wn**2.d0

        allocate (dis(2*n),vel(2*n),acc(2*n),p(2*n))
        dis=0.d0
        vel=0.d0
        acc=0.d0

        indat=10
        outdat=20

        call daq

        dis(1)=0.d0
```

```
        vel(1)=0.d0
        acc(1)=(p(1)-c*vel(1)-k*dis(1))/m

        print*, "Choose method"
        print*, "1. Interpolation of excitation"
        print*, "2. Central difference method"
        print*, "3. Newmark method"
        print*, "4. Fast fourier transform"
        read*, choose

        if (choose.eq.1) call interpolation
        if (choose.eq.2) call central
        if (choose.eq.3) call newmark
        if (choose.eq.4) call fft

        close(indat)
        close(outdat)

end program hw_ch5

subroutine daq
        use form
        implicit none
        integer indat,outdat,i,n
        double precision,allocatable :: gacc(:)
        character(20) input

        n=2048
        indat=10
        outdat=20

        allocate (gacc(2*n))

        gacc = 0.d0

        print*, "Input name of data file! "
```

```
        read*, input
        open (indat, file=trim(input), status="old", action="read")
        open (outdat, file='output.lsh')

        do i=1,2*n
                if(i<1560) then
                read(indat,*) gacc(i)
                elseif (i>=1560) then
                gacc(i)=0.d0
                endif
                p(i)=-m*gacc(i)
        enddo

end subroutine daq

subroutine interpolation
        use form
        implicit none

        integer i,n
        double precision g1,g2,g3,g4,a1,b1,c1,d1,a2,b2,c2,d2

        n=1559

        g1=dexp(-xi*wn*dt)
        g2=xi/dsqrt(1.d0-xi**2)
        g3=dsin(wd*dt)
        g4=dcos(wd*dt)

        a1=g1*(g2*g3+ g4)
        b1=g1*(g3/wd)
        c1=((2.d0*xi)/(wn*dt)+ g1*(((1.d0-2.d0*xi**2.d0)/(wd*dt)-g2)*g3-&
                                        (1.d0+ (2.d0*xi)/(wn*dt))*g4))/k
        d1=(1.d0-(2.d0*xi)/(wn*dt)+ g1*((2.d0*xi**2.d0-1)/(wd*dt)*g3&
                                        + (2.d0*xi)/(wn*dt)*g4))/k
```

```
        a2=-g1*(wn/dsqrt(1.d0-xi**2)*g3)
        b2=g1*(g4-g2*g3)
        c2=(-1.d0/dt+ g1*((wn/dsqrt(1.d0-xi**2)+ g2/dt)*g3+ g4/dt))/k
        d2=(1.d0-g1*(g2*g3+ g4))/(k*dt)

        do i=1,n-1
                dis(i+ 1)=a1*dis(i)+ b1*vel(i)+ c1*p(i)+ d1*p(i+ 1)
                vel(i+ 1)=a2*dis(i)+ b2*vel(i)+ c2*p(i)+ d2*p(i+ 1)
                acc(i+ 1)=(p(i+ 1)-c*vel(i+ 1)-k*dis(i+ 1))/m
        enddo

end subroutine interpolation

subroutine central
        use form
        implicit none

        integer i,n
        double precision a,b,khat,phat,uhat,unp1

        n=1559

        uhat=dis(1)-dt*vel(1)+ dt**2.d0*acc(1)/2.d0
        khat=m/(dt**2.d0)+ c/(2.d0*dt)
        a=m/(dt**2.d0)-c/(2.d0*dt)
        b=k-2*m/(dt**2.d0)
        dis(2)=(p(1)-a*uhat-b*dis(1))/khat

        do i=2,n-1
                phat=p(i)-a*dis(i-1)-b*dis(i)
                dis(i+ 1)=phat/khat
                vel(i)=(dis(i+ 1)-dis(i-1))/2.d0*dt
                acc(i)=(dis(i+ 1)-2.d0*dis(i)+ dis(i-1))/dt**2.d0
        enddo

        unp1=(p(n)-a*dis(n-1)-b*dis(n))/khat
```

```
        vel(n)=(unp1-dis(n-1))/2.d0*dt
        acc(n)=(unp1-2.d0*dis(n)+ dis(n-1))/dt**2.d0

end subroutine central

subroutine newmark
        use form
        implicit none

        integer i,accopt,n
        double precision gam,bet,a,b,khat,ddis,dvel,dacc,dp

        print*, "Choose type of method"
        print*, "1. average acc method"
        print*, "2. linear acc method"
        read*, accopt

        n=1559

        if (accopt.eq.1) then
                gam=0.5d0
                bet=0.25d0
        elseif (accopt.eq.2) then
                gam=0.5d0
                bet=1.d0/6.d0
        endif

        khat=k+ gam/(bet*dt)*c+ 1.d0/(bet*dt**2)*m
        a=1.d0/(bet*dt)*m+ gam/bet*c
        b=1.d0/(2.d0*bet)*m+ dt*(gam/(2.d0*bet)-1.d0)*c

        do i=1,n-1
                dp=p(i+ 1)-p(i)
                dp=dp+ a*vel(i)+ b*acc(i)
                ddis=dp/khat
                dvel=gam/(bet*dt)*ddis-gam/bet*vel(i)+ dt*(1.d0-&
```

```
                                        gam/(2.d0*bet))*acc(i)
            dacc=1.d0/(bet*dt**2)*ddis-1.d0/(bet*dt)*vel(i)-1.d0/(2.d0*bet)*acc(i)
            dis(i+1)=dis(i)+ddis
            vel(i+1)=vel(i)+dvel
            acc(i+1)=acc(i)+dacc
        enddo
        call lshresult
end subroutine newmark


subroutine fft
        use form
        use imslf90
        implicit none

        integer i,n
        double precision T,dw
        double complex,allocatable :: disc(:),velc(:),accc(:),&
                                        pc(:),disf(:),velf(:),accf(:),h(:),hf(:),pf(:)

        n=2048

        T=2.d0*n*dt
        dw=2.d0*pi/T
        allocate (disc(2*n),velc(2*n),accc(2*n),pc(2*n),disf(2*n),&
            velf(2*n),accf(2*n),hf(2*n),pf(2*n))

        do i=1,2*n
            if (i<=n) then
                pc(i)=dcmplx(p(i),0.d0)
            elseif (i>n) then
                pc(i)=0.d0
            endif
        enddo

        do i=1,2*n
```

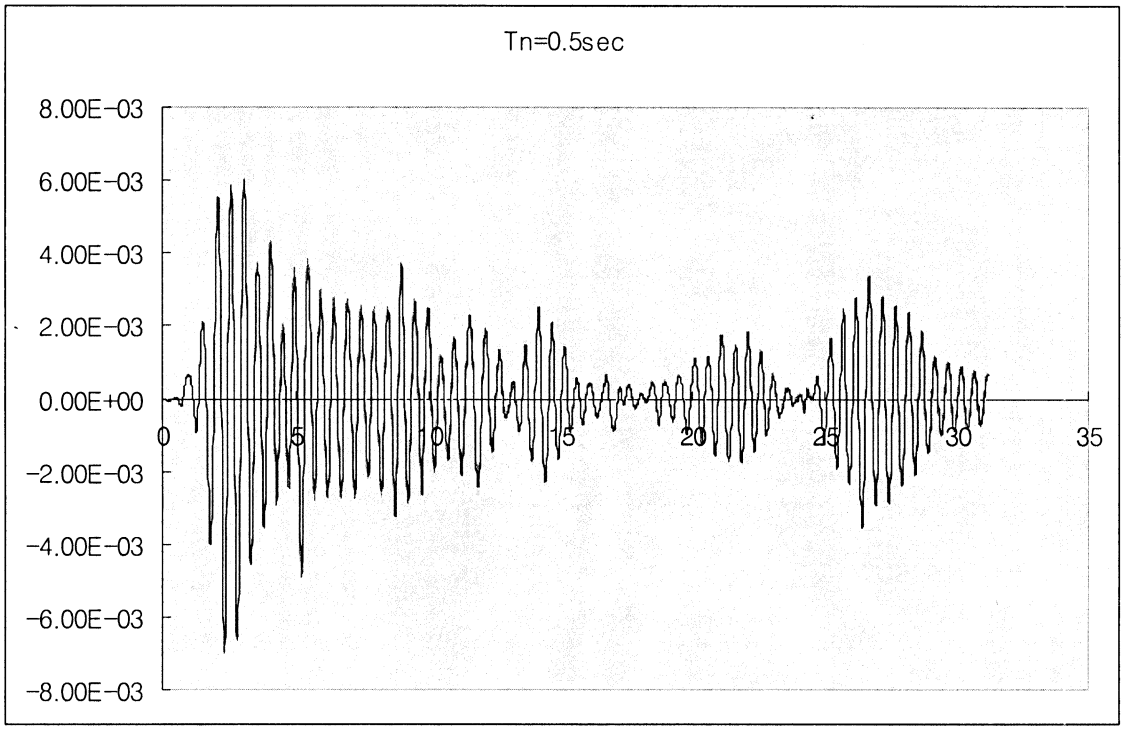
```
        hf(i)=1/dcmplx(k-m*((i-1)*dw)**2.d0,(i-1)*dw*c)
enddo

call dfftcf(2*n,pc,pf)
pf=pf*dt

disf(1)=0.d0
disf(n+1)=pf(n+1) * hf(n+1)
do i=2,n
        disf(i)=pf(i)*hf(i)
        disf(2*n+2-i) = dconjg(disf(i))
enddo

do i=2,n
        velf(i)=disf(i)*dcmplx(0.d0,(i-1)*dw)
        velf(2*n+2-i)=dconjg(velf(i))
enddo
velf(1)=0.d0
velf(n+1)=disf(n+1)*dcmplx(0.d0,n*dw)

do i=2,n
        accf(i)=(-((i-1)*dw)**2)*disf(i)
        accf(2*n+2-i)=dconjg(accf(i))
enddo
accf(1)=0.d0
accf(n+1)=disf(n+1)*dcmplx(-(n*dw)**2.d0,0.d0)

call dfftcb(2*n,disf,disc)
call dfftcb(2*n,velf,velc)
call dfftcb(2*n,accf,accc)

do i=2,n
        dis(i)=dreal(disc(i))/(2*n*dt)
        vel(i)=dreal(velc(i))/(2*n*dt)
        acc(i)=dreal(accc(i))/(2*n*dt)
enddo
```

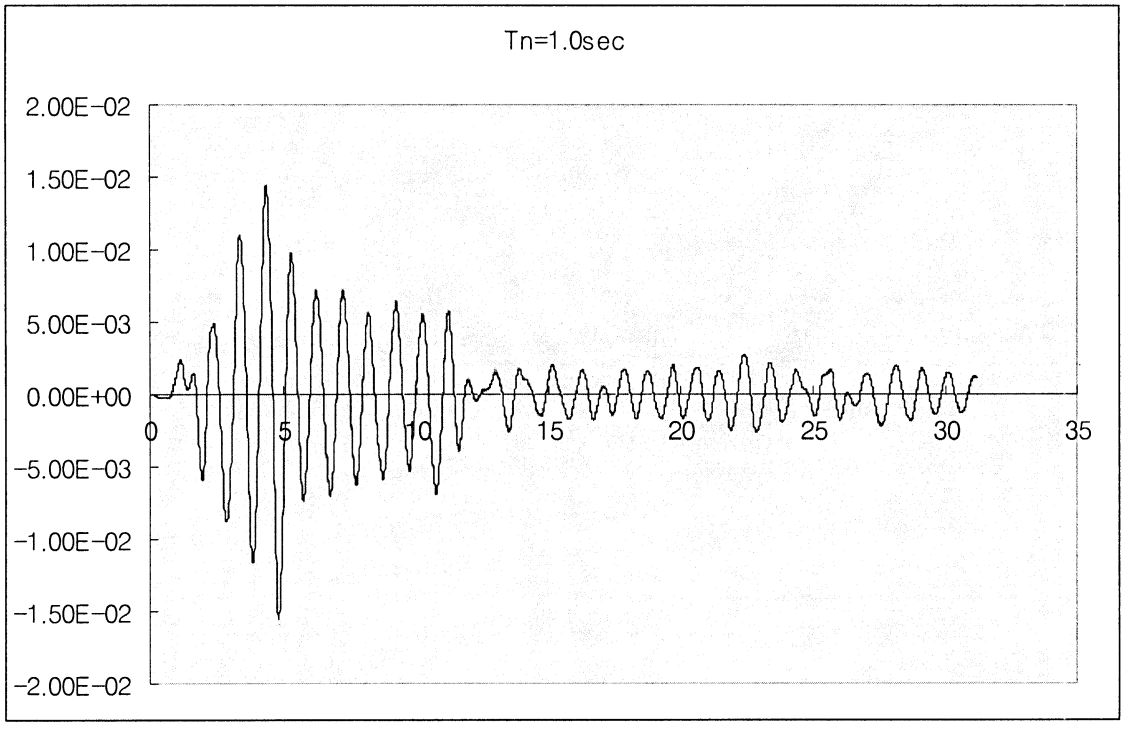
```
        call fftresult
end subroutine fft

subroutine lshresult
        use form
        implicit none
        integer outdat,i,n

        outdat=20
        n=1559

        write(outdat,'(/)')
        write(outdat,'(1x,a7,f3.1,a4)') 'Tn is= ',Tn,' sec'
        write(outdat,'(/)')
        write(outdat,'(1x,a65)') '    Time          Displacement    &
                                          Velocity          Acceleration '
        do i=1,n
                write(outdat,30) (i-1.d0)*dt,dis(i),vel(i),acc(i)
        enddo

        30 format(1x,f10.2,3e15.5)

end subroutine lshresult

subroutine fftresult
        use form
        implicit none
        integer outdat,i,n

        outdat=20
        n=2048

        write(outdat,'(/)')
        write(outdat,'(1x,a7,f3.1,a4)') 'Tn is= ',Tn,' sec'
        write(outdat,'(/)')
        write(outdat,'(1x,a65)') '    Time          Displacement     &
```

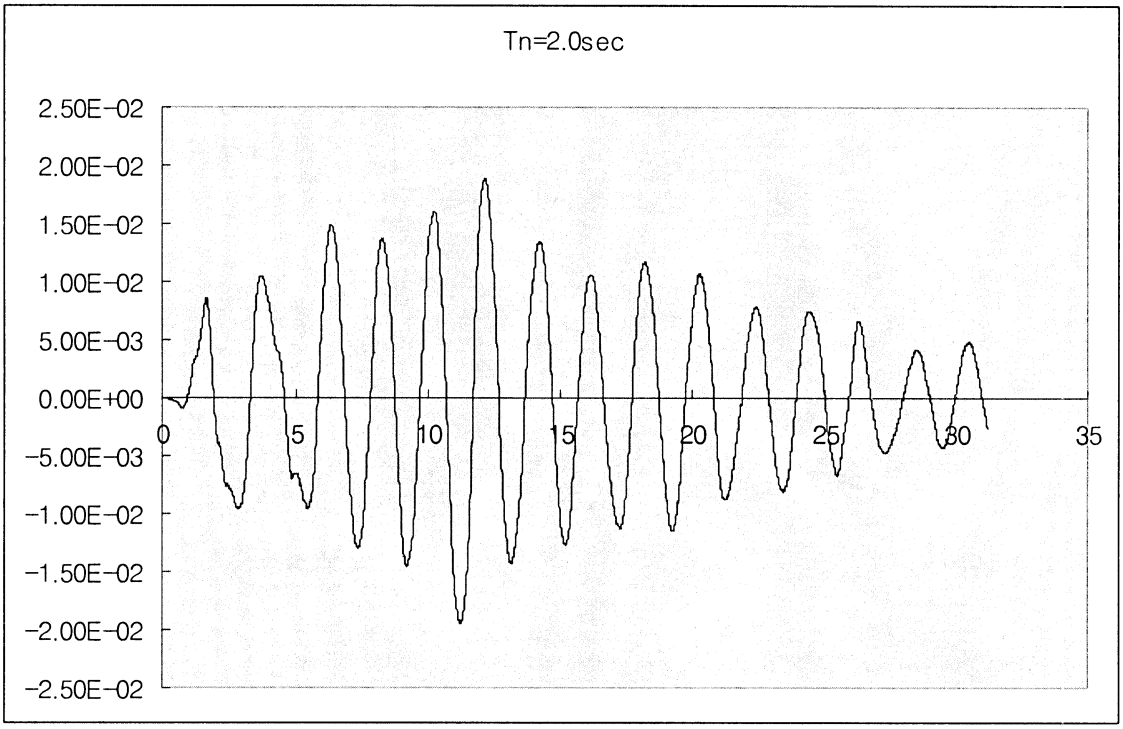
```
                                        Velocity          Acceleration '
      do i=1,n
            write(outdat,30) (i-1.d0)*dt,dis(i),vel(i),acc(i)
      enddo

      30 format(1x,f10.2,3e15.5)
end subroutine fftresult
```

2. Diagram

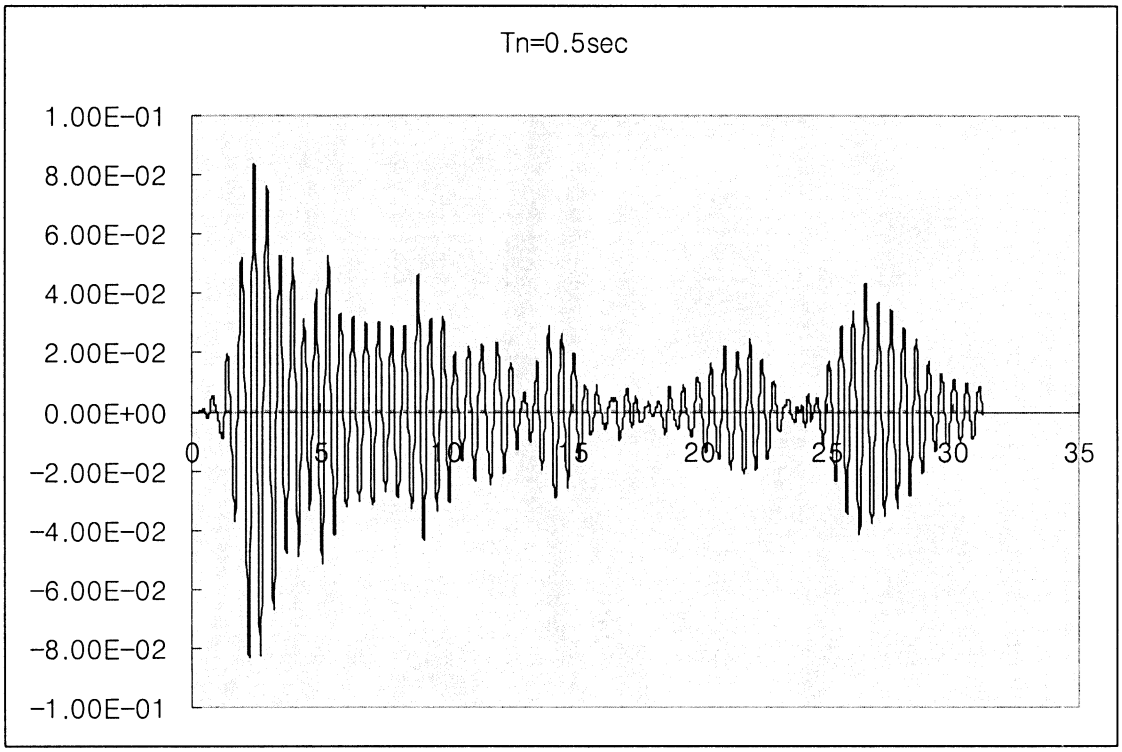
◈ *displacement*

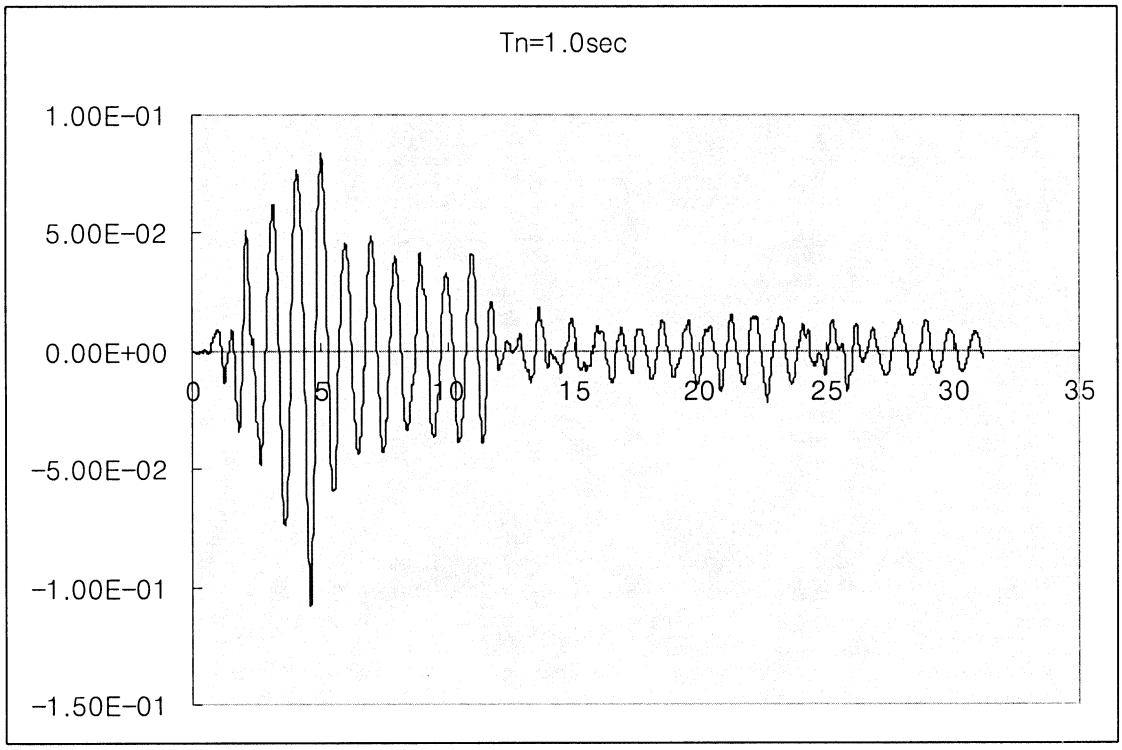


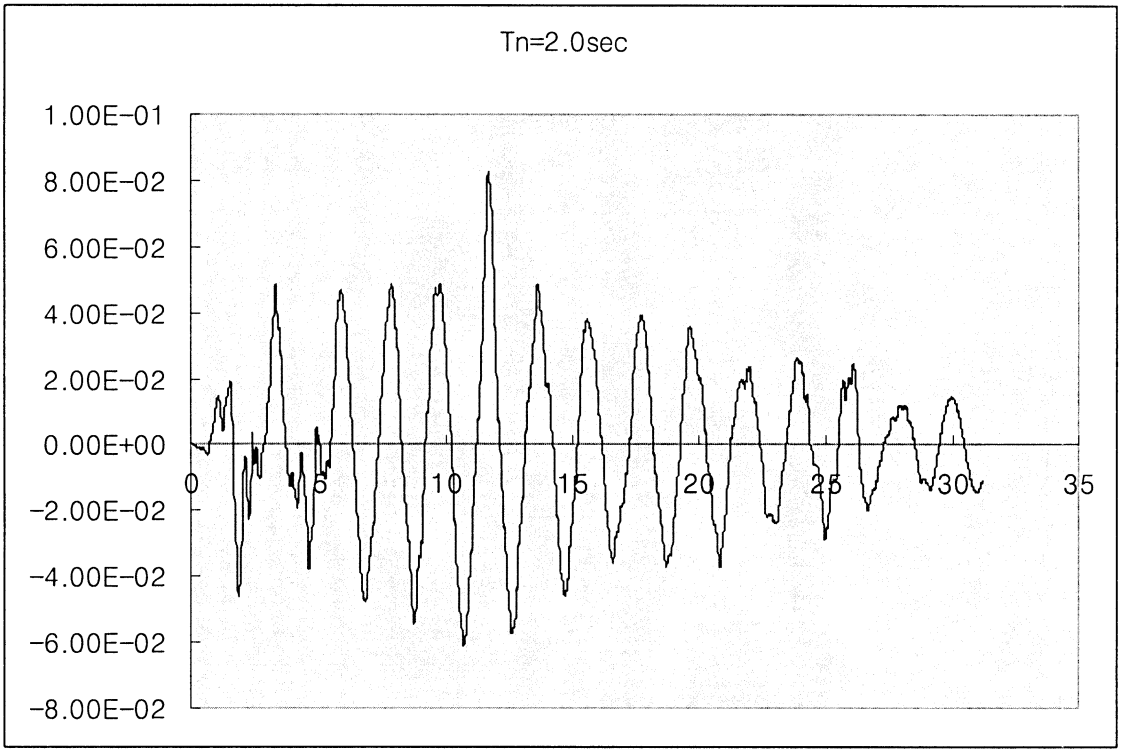



Tn=2.0sec
2.50E-02
2.00E-02
1.50E-02
1.00E-02
5.00E-03
0.00E+00
-5.00E-03
-1.00E-02
-1.50E-02
-2.00E-02
-2.50E-02
0
5
10
15
20
25
30
35

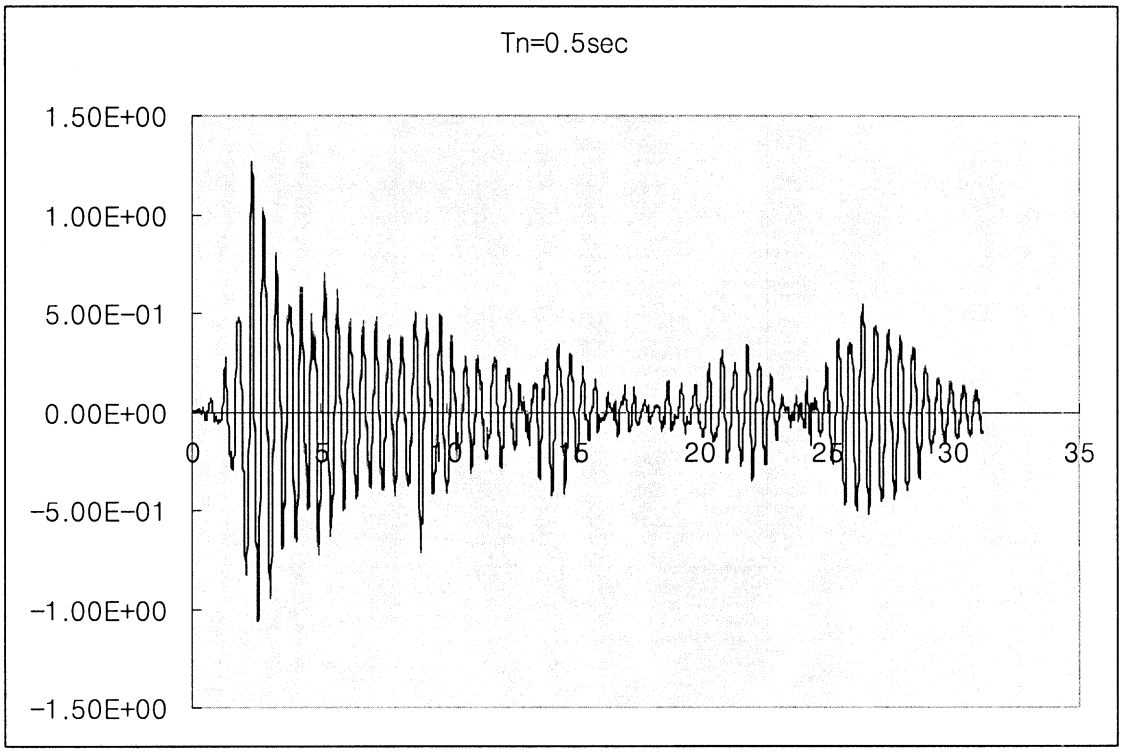
◈ *velocity*

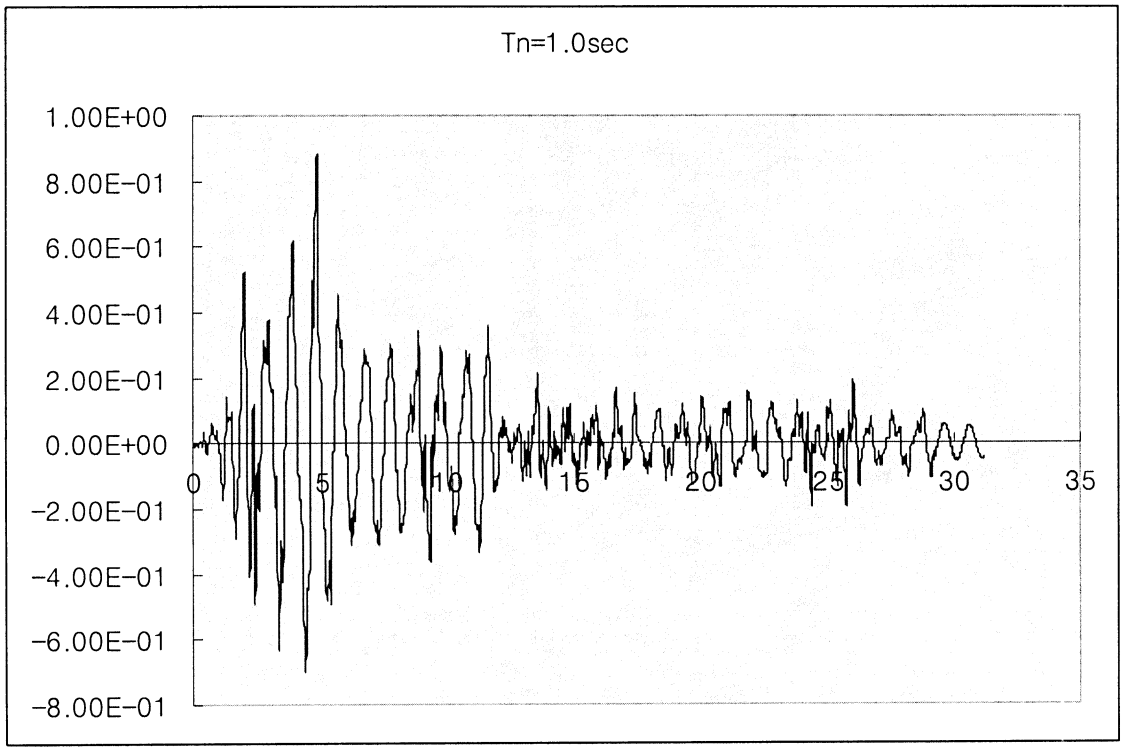

Tn=0.5sec
1.00E-01
8.00E-02
6.00E-02
4.00E-02
2.00E-02
0.00E+00
-2.00E-02
-4.00E-02
-6.00E-02
-8.00E-02
-1.00E-01
0
5
10
15
20
25
30
35

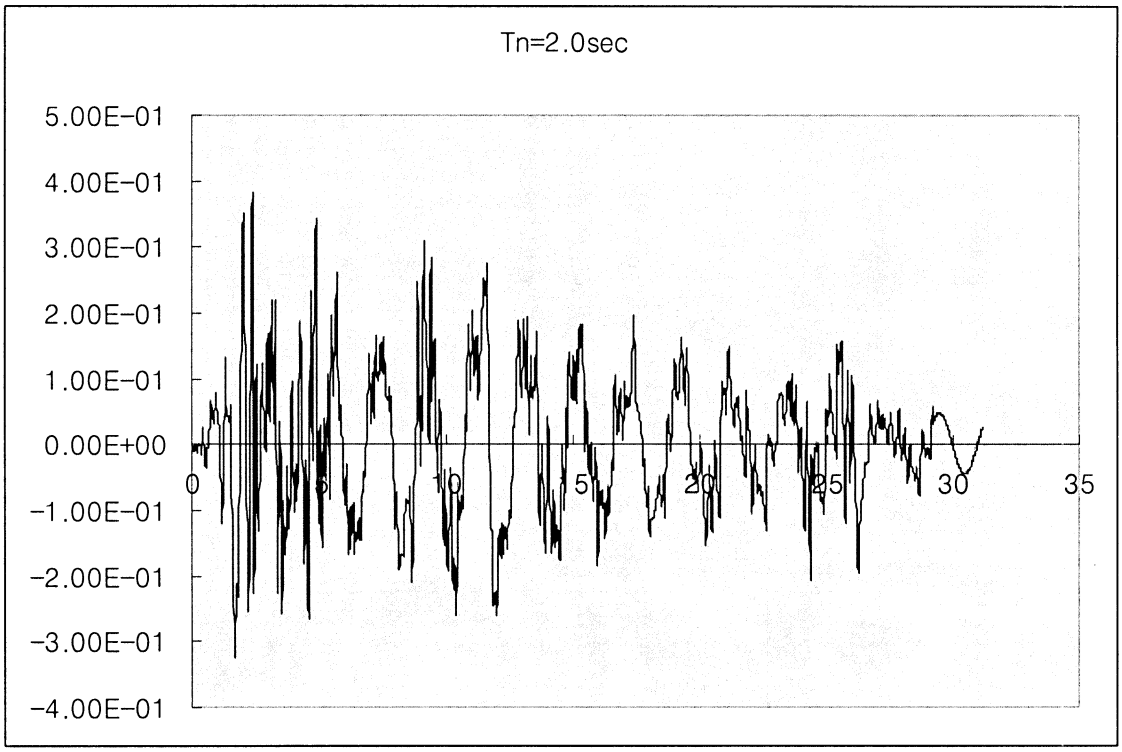

Tn=1.0sec
1.00E-01
5.00E-02
0.00E+00
-5.00E-02
-1.00E-01
-1.50E-01
0
5
10
15
20
25
30
35


Tn=2.0sec
1.00E-01
8.00E-02
6.00E-02
4.00E-02
2.00E-02
0.00E+00
-2.00E-02
-4.00E-02
-6.00E-02
-8.00E-02
0
5
10
15
20
25
30
35

◈ *acceleration*







3. Comment

$m=1$ 을 사용하여 수치해석을 실시하였다. 또한 Interpolation of excitation, Central difference method, Newmark method 는 주어진 지진가속도 자료의 개수인 1559개의 데이터를 바탕으로 수치해석을 하였고, Fast fourier transform 의 경우 $2^m = N$ 을 만족시키기 위해 자료의 개수를 2048개로 설정하고 가진 후의 변위를 고려하기 위해 관찰시간을 $2 \times N \times \Delta t$ 로 잡아 수치해석을 실시하였다. 또한 Fourier 변환과 역변환은 IMSL Library에 있는 함수를 사용했다. 그 결과 4가지 방법 모두 그래프에서 분간이 불가능할 정도로 거의 일치하는 것을 볼 수 있다.