OpenSeismoMatlab

Software for strong ground motion data processing

-

Contents

Help

Help section of the OpenSeismoMatlab functions

-

baselineCorr.m

Baseline correction of acceleration time history

-

BLKIN.m

Bilinear elastoplastic hysteretic model with elastic viscous damping

-

CDReSp.m

Constant Ductility Response Spectra

-

DRHA.m

Dynamic Response History Analysis

-

FASp.m

Fourier amplitude spectrum

-

HalfStep.m

Reproduce a signal with half time step

-

IDA.m

Incremental Dynamic Analysis

-

LEReSp.m

Linear Elastic Response Spectra

-

LIDA.m

Linear Implicit Dynamic Analysis

-

NLIDABLKIN.m

Non Linear Implicit Dynamic Analysis of a bilinear kinematic hardening

-

OpenSeismoMatlab.m

Seismic parameters and analyses of an acceleration time history

-

Examples

Examples of various OpenSeismoMatlab applications

-

example_baselineCorr

Perform baseline correction

-

example_CDReSp

Calculate constant ductility response spectra of an artificially generated acceleration time history

-

example_Constant_ductility_response_spectra

Calculate constant ductility response spectra of a suite of ground motions

-

example_Fourier_spectra

Calculate Fourier spectra of an acceleration time history

-

example_general

Test all functionalities of OpenSeismoMatlab in a single script

-

example_LEReSp

Calculate linear elastic response spectra of an artificially generated acceleration time history

-

example_Linear_elastic_response_spectra.m

Calculate the linear elastic response spectra of an earthquake suite

-

example_Spectra_comparison_1.m

Comparison of elastic and constant ductility response spectra for mu=1

-

example_Spectra_comparison_2.m

Comparison of constant ductility response spectra for mu=2

-

Verification

Validation of OpenSeismoMatlab results against the literature

verification_DRHA

Calculate linear dynamic response of a MDOF shear building

Reference: Chopra (2020)

-

verification_filter1

Verify the high pass Butterworth filter of OpenSeismoMatlab

Reference: Boore (2005)

-

verification_filter2

Verify the low- and high- pass Butterworth filter of OpenSeismoMatlab

Reference: Graizer (2012)

-

verification_Fourier_spectrum1

Verify the Fourier amplitude spectrum of the San Fernando earthquake (1971) - Component N11E

Reference: California Institute of Technology, EERL (1974)

-

verification_Fourier_spectrum2

Verify the Fourier amplitude spectrum of the San Fernando earthquake (1971) - Component N79W

Reference: California Institute of Technology, EERL (1974)

-

verification_IDA1

Verify the incremental dynamic analysis for an acceleration time history with given duration D_5_75 and spectral acceleration

Reference: Mashayekhi et al. (2020)

-

verification_IDA2

Verify the incremental dynamic analysis for an earthquake suite

Reference: Deng et al. (2017)

-

verification_IDA3

Verify the incremental dynamic analysis for the Loma Prieta earthquake (1989)

Reference: Vamvatsikos & Cornell (2002)

-

verification_IDA4

Verify the incremental dynamic analysis bias due to fitting of the capacity curve of a SDOF system with an elastoplastic bilinear fit according to FEMA-440

Reference: De Luca et al. (2011)

-

verification_LIDA

Calculate the dynamic response of a linear SDOF oscillator

Reference: Chopra (2020)

-

verification_NLIDABLKIN

Verify the energy time history of SDOF oscillator

Reference: Chopra (2020)

-

Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

baselineCorr

Documentation of the baselineCorr function.

```
helpFun('baselineCorr')
```

```
  Baseline correction of acceleration time history

  [COR_XG,COR_XGT,COR_XGTT] = BASELINECORR(T,XGTT)

  Description
      Linear baseline correction is performed for an uncorrected
      acceleration time history. Initially, first order fitting (straight
      line) is performed on the acceleration time history and the fitting
      line is subrtacted from the acceleration time history, giving thus
      the first correction. Afterwards, the first correction of the
      acceleration is integrated to obtain the velocity, and then first
      order fitting (straight line) is performed on this velocity time
      history. The gradient of the straight fitting line is then subtracted
      from the first correction of the acceleration time history, giving
      thus the second correction of the acceleration time history. The
      second correction of the acceleration time history is then integrated
      to give the corrected velocity and displacement time histories.

  Input parameters
      T [double(1:numsteps x 1)] is the time vector of the input
          acceleration time history XGTT. numsteps is the length of the
          input acceleration time history.
      XGTT [double(1:nstep x 1)]: column vector of the acceleration history
          of the excitation imposed at the base. nstep is the number of
          time steps of the dynamic response.

  Output parameters
      COR_XG [double(1:nstep x 1)]: time-history of displacement
      COR_XGT [double(1:nstep x 1)]: time-history of velocity
      COR_XGTT [double(1:nstep x 1)]: time-history of acceleration

  Example
      fid=fopen('elcentro.dat','r');
      text=textscan(fid,'%f %f');
      fclose(fid);
      time=text{1,1};
      xgtt1=text{1,2};
      dt=time(2)-time(1);
      xgt1 = cumtrapz(time,xgtt1);
      xg1 = cumtrapz(time,xgt1);
      [xg2, xgt2, xgtt2] = baselineCorr(time,xgtt1)
      figure()
      plot(time,xgtt1)
      hold on
      plot(time,xgtt2)
      figure()
      plot(time,xgt1)
      hold on
      plot(time,xgt2)
      figure()
```

```
    plot(time,xg1)
    hold on
    plot(time,xg2)
```


```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```


Published with MATLAB® R2017b

BLKIN

Documentation of the BLKIN function.

```
helpFun('BLKIN')
```

```
  Bilinear kinematic hysteretic model with elastic viscous damping

  [F,K,C,K_STATUS,D] = BLKIN(U,UT,K_HI,K_LO,UY,M,KSI,K_STATUS,D)

  Description
      Define the internal force vector, tangent stiffness matrix and
      tangent damping matrix of a bilinear elastoplastic hysteretic
      structure with elastic damping as a function of displacement and
      velocity.
      The MDOF structure modeled with this function consists of lumped
      masses connected with stiffness and damping elements in series. Each
      lumped mass has one degree of freedom. The first degree of freedom is
      at the top of the structure and the last at its fixed base. However,
      the last degree of freedom is not included in the input arguments of
      the function, i.e. not contained in ndof, as it is always fixed.
      The nonlinear stiffness is virtually of the bilinear type, where an
      initial stiffness and a post-yield stiffness are defined. The
      unloading or reloading curve of this model are parallel to the
      initial loading curve, and a hysteresis loop is created by
      continuously loading and unloading the structure above its yield
      limit. This behavior can be viewed as hardening of the kinematic
      type.
      An appropriate reference for this function definition is Hughes,
      Pister & Taylor (1979): "Implicit-explicit finite elements in
      nonlinear transient analysis". This function should be defined in
      accordance with equations (3.1), (3.2) and (3.3) of this paper. This
      representation has as special cases nonlinear elasticity and a class
      of nonlinear “rate-type” viscoelastic materials. Tangent stiffness
      and tangent damping matrices are the "consistent" linearized
      operators associated to f in the sense of [Hughes & Pister,
      "Consistent linearization in mechanics of solids", Computers and
      Structures, 8 (1978) 391-397].

  Input parameters
      U [double(1 x 1)] is the absolute displacement.
      UT [double(1 x 1)] is the absolute velocity.
      K_HI [double(1 x 1)] is the initial stiffness of the system before
          its first yield, i.e. the high stiffness.
      K_LO [double(1 x 1)] is the post-yield stiffness of the system, i.e.
          the low stiffness.
      UY [double(1 x 1)] is the yield limit of the structure. The structure
          is considered to yield, if the displacement exceeds uy(i).
      M [double(1 x 1)] is the lumped mass.
      KSI [double(1 x 1)] is the ratio of critical viscous damping of the
          system, assumed to be unique for all damping elements of the
          structure.
      K_STATUS [double(1 x 1)] is the is the stiffness vector which takes
          into account any plastic response of the structure. It is used to
          record the status of the structure so that it is known before the
          next application of this function at a next (time) step.
```

```
        Initialize by setting K_STATUS=K_HI.
    D [double(1 x 1)] is the is the equilibrium displacement vector which
        takes into account any plastic response of the structure. It is
        used to record the status of the structure so that it is known
        before the next application of this function at a next (time)
        step. Initialize by setting D=zeros(ndof,1).

Output parameters
    F [double(1 x 1)] is the internal force vector of the structure (sum
        of forces due to stiffness and damping) at displacement u and
        velocity ut
    K [double(1 x 1)] is the tangent stiffness matrix (nonlinear function
        of displacement u and velocity ut). It is equivalent to the
        derivative d(f)/d(u)
    C [double(1 x 1)] is the tangent damping matrix (nonlinear function
        of displacement u and velocity ut). It is equivalent to the
        derivative d(f)/d(u)
    K_STATUS [double(1 x 1)] is the is the stiffness vector which takes
        into account any plastic response of the structure. It is used to
        record the status of the structure so that it is known before the
        next application of this function at a next (time) step.
    D [double(1 x 1)] is the is the equilibrium displacement vector which
        takes into account any plastic response of the structure. It is
        used to record the status of the structure so that it is known
        before the next application of this function at a next (time)
        step.

Example
    u=0:0.2:4;
    u=[u,u(end:-1:1)];
    u=[u,-u];
    u=[u u];
    ut=0.001*ones(1,numel(u));
    ut=[ut,-ut];
    ut=[ut,ut(end:-1:1)];
    ut=[ut ut];
    k_hi=1000;
    k_lo=1;
    uy=2;
    M=1;
    ksi=0.05;
    k=k_hi;
    d=0;
    f=zeros(1,numel(u));
    for i=1:numel(u)
        [f(i),K,C,k,d] = BLKIN(u(i),ut(i),k_hi,k_lo,uy,M,ksi,k,d);
    end
    figure()
    plot(u,f)
```

```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```

Published with MATLAB® R2017b

CDReSp

Documentation of the CDReSp function.

```
helpFun('CDReSp')
```

```
  Constant Ductility Response Spectra

  [PSA,PSV,SD,SV,SA,FYK,MUK,ITERK]=CDRESP(DT,XGTT,T,KSI,MU,N,TOL,...
      PYSF,DTTOL,ALGID,RINF,MAXTOL,JMAX,DAK)

  Description
      The constant ductility response spectra for a given time-history of
      constant time step, a given eigenperiod range, a given viscous
      damping ratio and a given ductility are computed. See section 7.5 in
      Chopra (2012) and the notes "Inelastic Response Spectra" (CEE 541.
      Structural Dynamics) by Henri P. Gavin.

  Input parameters
      DT [double(1 x 1)] is the time step of the input acceleration time
          history XGTT.
      XGTT [double(1:numsteps x 1)] is the input acceleration time history.
          numsteps is the length of the input acceleration time history.
      T [double(1:numSDOFs x 1)] contains the values of eigenperiods for
          which the response spectra are requested. numSDOFs is the number
          of SDOF oscillators being analysed to produce the spectra.
      KSI [double(1 x 1)] is the fraction of critical viscous damping.
      MU [double(1 x 1)] is the target ductility for which the response
          spectra are calculated.
      N [double(1 x 1)] is the maximum number of iterations that can be
          performed until convergence of the calculated ductility to the
          target ductility is achieved.
      TOL [double(1 x 1)] is the tolerance for convergence for the target
          ductility.
      PYSF [double(1 x 1)] is the post-yield stiffness factor, i.e. the
          ratio of the postyield stiffness to the initial stiffness. PYSF=0
          is not recommended for simulation of an elastoplastic system. A
          small positive value is always suggested. PYSF is ignored if
          MU=1.
      DTTOL [double(1 x 1)] is the tolerance for resampling of the input
          acceleration time history. For a given eigenperiod T, resampling
          takes place if DT/T>dtTol.
      ALGID [char(1 x :inf)] is the algorithm to be used for the time
          integration. It can be one of the following strings for superior
          optimally designed algorithms:
              'generalized a-method': The generalized a-method (Chung &
              Hulbert, 1993)
              'HHT a-method': The Hilber-Hughes-Taylor method (Hilber,
              Hughes & Taylor, 1977)
              'WBZ': The Wood–Bossak–Zienkiewicz method (Wood, Bossak &
              Zienkiewicz, 1980)
              'U0-V0-Opt': Optimal numerical dissipation and dispersion
              zero order displacement zero order velocity algorithm
              'U0-V0-CA': Continuous acceleration (zero spurious root at
              the low frequency limit) zero order displacement zero order
              velocity algorithm
```

```
            'U0-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) zero order displacement zero order
            velocity algorithm
            'U0-V1-Opt': Optimal numerical dissipation and dispersion
            zero order displacement first order velocity algorithm
            'U0-V1-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) zero order displacement first order
            velocity algorithm
            'U0-V1-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) zero order displacement first order
            velocity algorithm
            'U1-V0-Opt': Optimal numerical dissipation and dispersion
            first order displacement zero order velocity algorithm
            'U1-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) first order displacement zero order
            velocity algorithm
            'U1-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) first order displacement zero order
            velocity algorithm
            'Newmark ACA': Newmark Average Constant Acceleration method
            'Newmark LA': Newmark Linear Acceleration method
            'Newmark BA': Newmark Backward Acceleration method
            'Fox-Goodwin': Fox-Goodwin formula
    RINF [double(1 x 1)] is the minimum absolute value of the eigenvalues
        of the amplification matrix. For the amplification matrix see
        eq.(61) in Zhou & Tamma (2004).
    MAXTOL [double(1 x 1)] is the maximum tolerance of convergence of the
        Full Newton Raphson method for numerical computation of
        acceleration.
    JMAX [double(1 x 1)] is the maximum number of iterations per
        increment. If JMAX=0 then iterations are not performed and the
        MAXTOL parameter is not taken into account.
    DAK [double(1 x 1)] is the infinitesimal acceleration for the
        calculation of the derivetive required for the convergence of the
        Newton-Raphson iteration.

Output parameters
    PSA [double(1:numSDOFs x 1)] is the Pseudo-Spectral Acceleration.
    PSV [double(1:numSDOFs x 1)] is the Pseudo-Spectral Velocity.
    SD [double(1:numSDOFs x 1)] is the Spectral Displacement.
    SV [double(1:numSDOFs x 1)] is the Spectral Velocity.
    SA [double(1:numSDOFs x 1)] is the Spectral Acceleration.
    FYK [double(1:numSDOFs x 1)] is the yield limit that each SDOF must
        have in order to attain ductility equal to muK.
    MUK [double(1:numSDOFs x 1)] is the achieved ductility for each
        period (each SDOF).
    ITERK [double(1:numSDOFs x 1)] is the number of iterations needed for
        convergence for each period (each SDOF).

Example
    dt=0.02;
    N=10;
    a=rand(N,1)-0.5;
    b=100*pi*rand(N,1);
    c=pi*(rand(N,1)-0.5);
    t=(0:dt:(100*dt))';
    xgtt=zeros(size(t));
    for i=1:N
        xgtt=xgtt+a(i)*sin(b(i)*t+c(i));
    end
    T=(0.04:0.04:4)';
```

```
    ksi=0.05;
    mu=2;
    n=50;
    tol=0.01;
    pysf=0.1;
    dtTol=0.02;
    AlgID='U0-V0-Opt';
    rinf=1;
    maxtol=0.01;
    jmax=200;
    dak=eps;
    [CDPSa,CDPSv,CDSd,CDSv,CDSa,fyK,muK,iterK]=CDReSp(dt,xgtt,T,ksi,...
        mu,n,tol,pysf,dtTol,AlgID,rinf,maxtol,jmax,dak);
    figure()
    plot(T,CDSd)
    figure()
    plot(T,fyK)
    figure()
    plot(T,muK)
    figure()
    plot(T,iterK)
```


```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```


Published with MATLAB® R2017b

DRHA

Documentation of the DRHA function.

```
helpFun('DRHA')
```

```
  Dynamic Response History Analysis

  [U,V,A,F,ES,ED] = DRHA(K,M,DT,XGTT,KSI,U0,UT0,ALGID,RINF)

  Description
      Calculate the dynamic response of a linear MDOF system using modal
      analysis. This function is part of the OpenSeismoMatlab software. It
      can be used as standalone, however attention is needed for the
      correctness of the input arguments, since no checks are performed in
      this function. See the example example_DRHA.m for more details about
      how this function can be implemented.

  Input parameters
      K [double(:inf x 1)] is the stiffness of the system.
      M [double(:inf x 1)] is the lumped masses of the structure.
      DT [double(1 x 1)] is the time step of the dynamic response history
          analysis
      XGTT [double(1:nstep x 1)]: column vector of the acceleration history
          of the excitation imposed at the base. nstep is the number of
          time steps of the dynamic response.
      KSI [double(1 x 1)] is the ratio of critical damping of the SDOF
          system.
      U0 [double(:inf x 1)] is the initial displacement of the SDOF system.
      UT0 [double(:inf x 1)] is the initial velocity of the SDOF system.
      ALGID [char(1 x :inf)] is the algorithm to be used for the time
          integration. It can be one of the following strings for superior
          optimally designed algorithms:
              'generalized a-method': The generalized a-method (Chung &
              Hulbert, 1993)
              'HHT a-method': The Hilber-Hughes-Taylor method (Hilber,
              Hughes & Taylor, 1977)
              'WBZ': The Wood–Bossak–Zienkiewicz method (Wood, Bossak &
              Zienkiewicz, 1980)
              'U0-V0-Opt': Optimal numerical dissipation and dispersion
              zero order displacement zero order velocity algorithm
              'U0-V0-CA': Continuous acceleration (zero spurious root at
              the low frequency limit) zero order displacement zero order
              velocity algorithm
              'U0-V0-DA': Discontinuous acceleration (zero spurious root at
              the high frequency limit) zero order displacement zero order
              velocity algorithm
              'U0-V1-Opt': Optimal numerical dissipation and dispersion
              zero order displacement first order velocity algorithm
              'U0-V1-CA': Continuous acceleration (zero spurious root at
              the low frequency limit) zero order displacement first order
              velocity algorithm
              'U0-V1-DA': Discontinuous acceleration (zero spurious root at
              the high frequency limit) zero order displacement first order
              velocity algorithm
              'U1-V0-Opt': Optimal numerical dissipation and dispersion
```

```
            first order displacement zero order velocity algorithm
            'U1-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) first order displacement zero order
            velocity algorithm
            'U1-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) first order displacement zero order
            velocity algorithm
            'Newmark ACA': Newmark Average Constant Acceleration method
            'Newmark LA': Newmark Linear Acceleration method
            'Newmark BA': Newmark Backward Acceleration method
            'Fox-Goodwin': Fox-Goodwin formula
    RINF [double(1 x 1)] is the minimum absolute value of the eigenvalues
        of the amplification matrix. For the amplification matrix see
        eq.(61) in Zhou & Tamma (2004).

Output parameters
    U [double(1 x 1:nstep)]: displacement time history.
    V [double(1 x 1:nstep)]: velocity time history.
    A [double(1 x 1:nstep)]: acceleration time history.
    F [double(1 x 1:nstep)]: equivalent static force time history.
    ES [double(1 x 1:nstep)]: time-history of the recoverable
        strain energy of the system (total and not incremental).
    ED [double(1 x 1:nstep)]: time-history of the energy
        dissipated by viscoelastic damping during each time step
        (incremental). cumsum(Ed) gives the time history of the total
        energy dissipated at dof i from the start of the dynamic
        analysis.
```

```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```

Published with MATLAB® R2017b

FASp

Documentation of the FASp function.

```
helpFun('FASp')
```

```
  Fourier amplitude spectrum

  [F,U] = FASP(DT,XGTT)

  Description
      Fourier amplitude spectrum of an acceleration time history.

  Input parameters
      DT [double(1 x 1)] is the time step of the input acceleration time
          history XGTT.
      XGTT [double(1:numsteps x 1)] is the input acceleration time history.
          numsteps is the length of the input acceleration time history.

  Output parameters
      F [double(1:2^(nextpow2(length(XGTT))-1) x 1)] is the frequency range
          in which the Fourier amplitudes are calculated.
      U [double(1:2^(nextpow2(length(XGTT))-1) x 1)] contains the Fourier
          amplitudes

  ____________________________________________________________________________
  Copyright (c) 2018-2022
      George Papazafeiropoulos
      Major, Infrastructure Engineer, Hellenic Air Force
      Civil Engineer, M.Sc., Ph.D.
      Email: gpapazafeiropoulos@yahoo.gr
  ____________________________________________________________________________
```

Published with MATLAB® R2017b

HalfStep

Documentation of the HalfStep function.

```
helpFun('HalfStep')
```

```
  Reproduce signal with half time step

  UNEW = HALFSTEP(U)

  Input parameters
      U [double(1:n x 1)] is the input signal with time step dt.

  Output parameters
      UNEW [double(1:n x 1)] is the output signal with time step dt/2.

  Verification:
      u=0.2:0.2:4;
      uNew=HalfStep(u);
      figure()
      plot((1:numel(u)),u)
      hold on
      plot((1:0.5:numel(u)),uNew)

 ____________________________________________________________________________
  Copyright (c) 2018-2022
      George Papazafeiropoulos
      Major, Infrastructure Engineer, Hellenic Air Force
      Civil Engineer, M.Sc., Ph.D.
      Email: gpapazafeiropoulos@yahoo.gr
  ____________________________________________________________________________
```

Published with MATLAB® R2017b

IDA

Documentation of the IDA function.

```
helpFun('IDA')
```

```
  Incremental Dynamic Analysis

  [DM,IM]=IDA(DT,XGTT,T,LAMBDAF,IM_DM,M,UY,PYSF,KSI,ALGID,U0,UT0,...
      RINF,MAXTOL,JMAX,DAK)

  Description
      This function performs incremental dynamic analysis of a given
      acceleration time history and SDOF oscillator.

  Input parameters
      DT [double(1 x 1)] is the time step of the input acceleration time
          history XGTT.
      XGTT [double(:inf x 1)] is the input acceleration time history.
          numsteps is the length of the input acceleration time history.
      T [double(1 x 1)] contains the eigenperiod of the SDOF system for
          which the incremental dynamic analysis response curve is
          requested.
      LAMBDAF [double(:inf x 1)] contains the values of the scaling factor
          (lambda factor) for the incremental dynamic analysis.
      IM_DM [char(1 x :inf)] is the Intensity Measure (IM) - Damage Measure
          (DM) pair that is to be calculated from the incremental dynamic
          analysis. IM_DM can take one of the following values (strings are
          case insensitive):
          'SA_MU': Spectral acceleration-ductility
          'PGD_MU': Peak displacement-ductility
          'PGV_MU': Peak velocity-ductility
          'PGA_MU': Peak acceleration-ductility
          'SA_DISP': Spectral acceleration-displacement
          'PGD_DISP': Peak displacement-displacement
          'PGV_DISP': Peak velocity-displacement
          'PGA_DISP': Peak acceleration-displacement
          'SA_VEL': Spectral acceleration-velocity
          'PGD_VEL': Peak displacement-velocity
          'PGV_VEL': Peak velocity-velocity
          'PGA_VEL': Peak acceleration-velocity
          'SA_ACC': Spectral acceleration-acceleration
          'PGD_ACC': Peak displacement-acceleration
          'PGV_ACC': Peak velocity-acceleration
          'PGA_ACC': Peak acceleration-acceleration
      M [double(1 x 1)] is the mass of the SDOF oscillator.
      UY [double(1 x 1)] is the yield displacement of the SDOF oscillator.
      PYSF [double(1 x 1)] is the post-yield stiffness factor, i.e. the
          ratio of the postyield stiffness to the initial stiffness. PYSF=0
          is not recommended for simulation of an elastoplastic system. A
          small positive value is always suggested. PYSF is ignored if
          MU=1.
      KSI [double(1 x 1)] is the fraction of critical viscous damping.
      ALGID [char(1 x :inf)] is the algorithm to be used for the time
          integration. It can be one of the following strings for superior
          optimally designed algorithms:
```

```
            'generalized a-method': The generalized a-method (Chung &
            Hulbert, 1993)
            'HHT a-method': The Hilber-Hughes-Taylor method (Hilber,
            Hughes & Taylor, 1977)
            'WBZ': The Wood–Bossak–Zienkiewicz method (Wood, Bossak &
            Zienkiewicz, 1980)
            'U0-V0-Opt': Optimal numerical dissipation and dispersion
            zero order displacement zero order velocity algorithm
            'U0-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) zero order displacement zero order
            velocity algorithm
            'U0-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) zero order displacement zero order
            velocity algorithm
            'U0-V1-Opt': Optimal numerical dissipation and dispersion
            zero order displacement first order velocity algorithm
            'U0-V1-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) zero order displacement first order
            velocity algorithm
            'U0-V1-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) zero order displacement first order
            velocity algorithm
            'U1-V0-Opt': Optimal numerical dissipation and dispersion
            first order displacement zero order velocity algorithm
            'U1-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) first order displacement zero order
            velocity algorithm
            'U1-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) first order displacement zero order
            velocity algorithm
            'Newmark ACA': Newmark Average Constant Acceleration method
            'Newmark LA': Newmark Linear Acceleration method
            'Newmark BA': Newmark Backward Acceleration method
            'Fox-Goodwin': Fox-Goodwin formula
    U0 [double(1 x 1)] is the initial displacement of the SDOF
        oscillator.
    UT0 [double(1 x 1)] is the initial velocity of the SDOF oscillator.
    RINF [double(1 x 1)] is the minimum absolute value of the eigenvalues
        of the amplification matrix. For the amplification matrix see
        eq.(61) in Zhou & Tamma (2004). Default value 0.
    MAXTOL [double(1 x 1)] is the maximum tolerance of convergence of the
        Full Newton Raphson method for numerical computation of
        acceleration.
    JMAX [double(1 x 1)] is the maximum number of iterations per
        increment. If JMAX=0 then iterations are not performed and the
        MAXTOL parameter is not taken into account.
    DAK [double(1 x 1)] is the infinitesimal acceleration for the
        calculation of the derivetive required for the convergence of the
        Newton-Raphson iteration.

Output parameters
    DM [double(:inf x 1)] is the Damage Measure.
    IM [double(:inf x 1)] is the Intensity Measure.

Example
    eqmotions={'elcentro'};
    data=load([eqmotions{1},'.dat']);
    t=data(:,1);
    dt=t(2)-t(1);
    xgtt=data(:,2);
    sw='ida';
```

```
    T=1;
    lambdaF=logspace(log10(0.001),log10(10),100);
    IM_DM='Sa_disp';
    m=1;
    uy = 0.082*9.81/(2*pi/T)^2;
    pysf=0.01;
    ksi=0.05;
    S5=OpenSeismoMatlab(dt,xgtt,sw,T,lambdaF,IM_DM,m,uy,pysf,ksi);
    figure()
    plot(S5.DM*1000,S5.IM/9.81,'k','LineWidth',1)
    grid on
    xlabel('Displacement (mm)')
    ylabel('Sa(T1,5%)[g]')
    xlim([0,200])
    ylim([0,0.7])
```


```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```


Published with MATLAB® R2017b

LEReSp

Documentation of the LEReSp function.

```
helpFun('LEReSp')
```

```
  Linear Elastic Response Spectra

  [PSA,PSV,SD,SV,SA,SIEVABS,SIEVREL]=LERESP(DT,XGTT,T,KSI,DTTOL,...
      ALGID,RINF)

  Description
      The linear elastic response spectra for a given time-history of
      constant time step, a given eigenperiod range and a given viscous
      damping ratio are computed. These spectra include the spectral
      acceleration, spectral velocity, spectral displacement,
      pseudoacceleration, pseudovelocity, absolute equivalent input energy
      velocity and relative equivalent input energy velocity. This function
      is part of the OpenSeismoMatlab software. It can be used as
      standalone, however attention is needed for the correctness of the
      input arguments, since no checks are performed in this function. See
      the example example_LEReSp.m for more details about how this function
      can be implemented.

  Input parameters
      DT [double(1 x 1)] is the time step of the input acceleration time
          history XGTT.
      XGTT [double(1:numsteps x 1)] is the input acceleration time history.
          numsteps is the length of the input acceleration time history.
      T [double(1:numSDOFs x 1)] contains the values of eigenperiods for
          which the response spectra are requested. numSDOFs is the number
          of SDOF oscillators being analysed to produce the spectra.
      KSI [double(1 x 1)] is the fraction of critical viscous damping.
      DTTOL [double(1 x 1)] is the maximum ratio of the integration time
          step to the eigenperiod.
      ALGID [char(1 x :inf)] is the algorithm to be used for the time
          integration. It can be one of the following strings for superior
          optimally designed algorithms:
              'generalized a-method': The generalized a-method (Chung &
              Hulbert, 1993)
              'HHT a-method': The Hilber-Hughes-Taylor method (Hilber,
              Hughes & Taylor, 1977)
              'WBZ': The Wood–Bossak–Zienkiewicz method (Wood, Bossak &
              Zienkiewicz, 1980)
              'U0-V0-Opt': Optimal numerical dissipation and dispersion
              zero order displacement zero order velocity algorithm
              'U0-V0-CA': Continuous acceleration (zero spurious root at
              the low frequency limit) zero order displacement zero order
              velocity algorithm
              'U0-V0-DA': Discontinuous acceleration (zero spurious root at
              the high frequency limit) zero order displacement zero order
              velocity algorithm
              'U0-V1-Opt': Optimal numerical dissipation and dispersion
              zero order displacement first order velocity algorithm
              'U0-V1-CA': Continuous acceleration (zero spurious root at
              the low frequency limit) zero order displacement first order
```

```
            velocity algorithm
            'U0-V1-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) zero order displacement first order
            velocity algorithm
            'U1-V0-Opt': Optimal numerical dissipation and dispersion
            first order displacement zero order velocity algorithm
            'U1-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) first order displacement zero order
            velocity algorithm
            'U1-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) first order displacement zero order
            velocity algorithm
            'Newmark ACA': Newmark Average Constant Acceleration method
            'Newmark LA': Newmark Linear Acceleration method
            'Newmark BA': Newmark Backward Acceleration method
            'Fox-Goodwin': Fox-Goodwin formula
    RINF [double(1 x 1)] is the minimum absolute value of the eigenvalues
        of the amplification matrix. For the amplification matrix see
        eq.(61) in Zhou & Tamma (2004).

Output parameters
    PSA [double(1:numSDOFs x 1)] is the Pseudo Acceleration Spectrum.
    PSV [double(1:numSDOFs x 1)] is the Pseudo Velocity Spectrum.
    SD [double(1:numSDOFs x 1)] is the Spectral Displacement.
    SV [double(1:numSDOFs x 1)] is the Spectral Velocity.
    SA [double(1:numSDOFs x 1)] is the Spectral Acceleration.
    SIEVABS [double(1:numSDOFs x 1)] is the equivalent absolute input
        energy velocity.
    SIEVREL [double(1:numSDOFs x 1)] is the equivalent relative input
        energy velocity.

Example
    dt=0.02;
    N=10;
    a=rand(N,1)-0.5;
    b=100*pi*rand(N,1);
    c=pi*(rand(N,1)-0.5);
    t=(0:dt:(100*dt))';
    xgtt=zeros(size(t));
    for i=1:N
        xgtt=xgtt+a(i)*sin(b(i)*t+c(i));
    end
    T=logspace(log10(0.02),log10(50),1000)';
    ksi=0.05;
    dtTol=0.02;
    AlgID='U0-V0-Opt';
    rinf=1;
    [PSa,PSv,Sd,Sv,Sa,SievABS,SievREL]=LEReSp(dt,xgtt,T,ksi,dtTol,...
        AlgID,rinf);
```

```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```

Published with MATLAB® R2017b

LIDA

Documentation of the LIDA function.

```
helpFun('LIDA')
```

```
  Linear Implicit Dynamic Analysis

  [U,UT,UTT] = LIDA(DT,XGTT,OMEGA,KSI,U0,UT0,ALGID,RINF)

  Description
      Linear implicit direct time integration of second order differential
      equation of motion of dynamic response of linear elastic SDOF systems
      The General Single Step Single Solve (GSSSS) family of algorithms
      published by X.Zhou & K.K.Tamma (2004) is employed for direct time
      integration of the general linear or nonlinear structural Single
      Degree of Freedom (SDOF) dynamic problem. The optimal numerical
      dissipation and dispersion zero order displacement zero order
      velocity algorithm designed according to the above journal article,
      is used in this routine. This algorithm encompasses the scope of
      Linear Multi-Step (LMS) methods and is limited by the Dahlquist
      barrier theorem (Dahlquist,1963). The force - displacement - velocity
      relation of the SDOF structure is linear. This function is part of
      the OpenSeismoMatlab software. It can be used as standalone, however
      attention is needed for the correctness of the input arguments, since
      no checks are performed in this function. See the example
      example_LIDA.m for more details about how this function can be
      implemented.

  Input parameters
      DT [double(1 x 1)] is the time step
      XGTT [double(1:nstep x 1)] is the column vector of the acceleration
          history of the excitation imposed at the base. nstep is the
          number of time steps of the dynamic response.
      OMEGA [double(1 x 1)] is the eigenfrequency of the structure in
          rad/sec.
      KSI [double(1 x 1)] is the ratio of critical damping of the SDOF
          system.
      U0 [double(1 x 1)] is the initial displacement of the SDOF system.
      UT0 [double(1 x 1)] is the initial velocity of the SDOF system.
      ALGID [char(1 x :inf)] is the algorithm to be used for the time
          integration. It can be one of the following strings for superior
          optimally designed algorithms:
              'generalized a-method': The generalized a-method (Chung &
              Hulbert, 1993)
              'HHT a-method': The Hilber-Hughes-Taylor method (Hilber,
              Hughes & Taylor, 1977)
              'WBZ': The Wood–Bossak–Zienkiewicz method (Wood, Bossak &
              Zienkiewicz, 1980)
              'U0-V0-Opt': Optimal numerical dissipation and dispersion
              zero order displacement zero order velocity algorithm
              'U0-V0-CA': Continuous acceleration (zero spurious root at
              the low frequency limit) zero order displacement zero order
              velocity algorithm
              'U0-V0-DA': Discontinuous acceleration (zero spurious root at
              the high frequency limit) zero order displacement zero order
```

```
            velocity algorithm
            'U0-V1-Opt': Optimal numerical dissipation and dispersion
            zero order displacement first order velocity algorithm
            'U0-V1-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) zero order displacement first order
            velocity algorithm
            'U0-V1-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) zero order displacement first order
            velocity algorithm
            'U1-V0-Opt': Optimal numerical dissipation and dispersion
            first order displacement zero order velocity algorithm
            'U1-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) first order displacement zero order
            velocity algorithm
            'U1-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) first order displacement zero order
            velocity algorithm
            'Newmark ACA': Newmark Average Constant Acceleration method
            'Newmark LA': Newmark Linear Acceleration method
            'Newmark BA': Newmark Backward Acceleration method
            'Fox-Goodwin': Fox-Goodwin formula
    RINF [double(1 x 1)] is the minimum absolute value of the eigenvalues
        of the amplification matrix. For the amplification matrix see
        eq.(61) in Zhou & Tamma (2004).

Output parameters
    U [double(1:nstep x 1)] is the time-history of displacement
    UT [double(1:nstep x 1)] is the time-history of velocity
    UTT [double(1:nstep x 1)] is the time-history of acceleration

Example (Figure 6.6.1 in Chopra, Tn=1sec)
    dt=0.02;
    fid=fopen('elcentro.dat','r');
    text=textscan(fid,'%f %f');
    fclose(fid);
    xgtt=text{1,2};
    Tn=1;
    omega=2*pi/Tn;
    ksi=0.02;
    u0=0;
    ut0=0;
    AlgID='U0-V0-Opt';
    rinf=1;
    [u,ut,utt] = LIDA(dt,xgtt,omega,ksi,u0,ut0,AlgID,rinf);
    D=max(abs(u))/0.0254
```


```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```


Published with MATLAB® R2017b

NLIDABLKIN

Documentation of the NLIDABLKIN function.

```
helpFun('NLIDABLKIN')
```

```
Non Linear Implicit Dynamic Analysis of a bilinear kinematic hardening
hysteretic structure with elastic damping

[U,UT,UTT,FS,EY,ES,ED,JITER] = NLIDABLKIN(DT,XGTT,M,K_HI,K_LO,UY,...
    KSI,ALGID,U0,UT0,RINF,MAXTOL,JMAX,DAK)

Description
    General linear implicit direct time integration of second order
    differential equations of a bilinear elastoplastic hysteretic SDOF
    dynamic system with elastic damping, with lumped mass.
    The General Single Step Single Solve (GSSSS) family of algorithms
    published by X.Zhou & K.K.Tamma (2004) is employed for direct time
    integration of the general linear or nonlinear structural Single
    Degree of Freedom (SDOF) dynamic problem. Selection among 9
    algorithms, all designed according to the above journal article, can
    be made in this routine. These algorithms encompass the scope of
    Linear Multi-Step (LMS) methods and are limited by the Dahlquist
    barrier theorem (Dahlquist,1963).

Input parameters
    DT [double(1 x 1)] is the time step of the integration
    XGTT [double(1:NumSteps x 1)] is the acceleration time history which
        is imposed at the lumped mass of the SDOF structure.
    M [double(1 x 1)] is the lumped masses of the structure. Define the
        lumped masses from the top to the bottom, excluding the fixed dof
        at the base
    K_HI [double(1 x 1)] is the initial stiffness of the system before
        its first yield, i.e. the high stiffness. Give the stiffness of
        each storey from top to bottom.
    K_LO [double(1 x 1)] is the post-yield stiffness of the system,
        i.e. the low stiffness. Give the stiffness of each storey from
        top to bottom.
    UY [double(1 x 1)] is the yield limit of the stiffness elements of
        the structure. The element is considered to yield, if the
        interstorey drift between degrees of freedom i and i+1 exceeds
        UY(i). Give the yield limit of each storey from top to bottom.
    KSI [double(1 x 1)] is the ratio of critical viscous damping of the
        system, assumed to be unique for all damping elements of the
        structure.
    ALGID [char(1 x :inf)] is the algorithm to be used for the time
        integration. It can be one of the following strings for superior
        optimally designed algorithms:
            'generalized a-method': The generalized a-method (Chung &
            Hulbert, 1993)
            'HHT a-method': The Hilber-Hughes-Taylor method (Hilber,
            Hughes & Taylor, 1977)
            'WBZ': The Wood–Bossak–Zienkiewicz method (Wood, Bossak &
            Zienkiewicz, 1980)
            'U0-V0-Opt': Optimal numerical dissipation and dispersion
            zero order displacement zero order velocity algorithm
```

```
            'U0-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) zero order displacement zero order
            velocity algorithm
            'U0-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) zero order displacement zero order
            velocity algorithm
            'U0-V1-Opt': Optimal numerical dissipation and dispersion
            zero order displacement first order velocity algorithm
            'U0-V1-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) zero order displacement first order
            velocity algorithm
            'U0-V1-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) zero order displacement first order
            velocity algorithm
            'U1-V0-Opt': Optimal numerical dissipation and dispersion
            first order displacement zero order velocity algorithm
            'U1-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) first order displacement zero order
            velocity algorithm
            'U1-V0-DA': Discontinuous acceleration (zero spurious root at
            the high frequency limit) first order displacement zero order
            velocity algorithm
            'Newmark ACA': Newmark Average Constant Acceleration method
            'Newmark LA': Newmark Linear Acceleration method
            'Newmark BA': Newmark Backward Acceleration method
            'Fox-Goodwin': Fox-Goodwin formula
    U0 [double(1 x 1)] is the initial displacement.
    UT0 [double(1 x 1)] is the initial velocity.
    RINF [double(1 x 1)] is the minimum absolute value of the eigenvalues
        of the amplification matrix. For the amplification matrix see
        eq.(61) in Zhou & Tamma (2004).
    MAXTOL [double(1 x 1)] is the maximum tolerance of convergence of the
        Full Newton Raphson method for numerical computation of
        acceleration.
    JMAX [double(1 x 1)] is the maximum number of iterations per
        increment. If JMAX=0 then iterations are not performed and the
        MAXTOL parameter is not taken into account.
    DAK [double(1 x 1)] is the infinitesimal acceleration for the
        calculation of the derivetive required for the convergence of the
        Newton-Raphson iteration.

Output parameters
    U [double(1 x 1:NumSteps)] is the time-history of displacement
    UT [double(1 x 1:NumSteps)] is the time-history of velocity
    UTT [double(1 x 1:NumSteps)] is the time-history of acceleration
    FS [double(1 x 1:NumSteps)] is the time-history of the internal
        force of the structure analysed.
    EY [double(1 x 1:NumSteps)] is the time history of the sum of the
        energy dissipated by yielding during each time step and the
        recoverable strain energy of the system (incremental).
        cumsum(EY)-Es gives the time history of the total energy
        dissipated by yielding from the start of the dynamic analysis.
    ES [double(1 x 1:NumSteps)] is the time-history of the recoverable
        strain energy of the system (total and not incremental).
    ED [double(1 x 1:NumSteps)] is the time-history of the energy
        dissipated by viscoelastic damping during each time step
        (incremental). cumsum(ED) gives the time history of the total
        energy dissipated from the start of the dynamic analysis.
    JITER [double(1 x 1:NumSteps)] is the iterations per increment

Notation in the code
```

```
    u=displacement
    un=displacement after increment n
    ut=velocity
    utn=velocity after increment n
    utt=acceleration
    uttn=acceleration after increment n
```


```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```


Published with MATLAB® R2017b

OpenSeismoMatlab

Documentation of the OpenSeismoMatlab function.

```
helpFun('OpenSeismoMatlab')
```

```
  Seismic parameters and processing of an acceleration time history

  Syntax
      PARAM=OpenSeismoMatlab(DT,XGTT,SW,__)
      PARAM=OpenSeismoMatlab(DT,XGTT,'PGA')
      PARAM=OpenSeismoMatlab(DT,XGTT,'PGV')
      PARAM=OpenSeismoMatlab(DT,XGTT,'PGD')
      PARAM=OpenSeismoMatlab(DT,XGTT,'ARIAS')
      PARAM=OpenSeismoMatlab(DT,XGTT,'TIMEHIST',BASELINESW)
      PARAM=OpenSeismoMatlab(DT,XGTT,'RESAMPLE',DTI)
      PARAM=OpenSeismoMatlab(DT,XGTT,'ES',T,KSI,ALGID,RINF,DTTOL)
      PARAM=OpenSeismoMatlab(DT,XGTT,'CDS',T,KSI,MU,PYSF,DTTOL,ALGID,...
          RINF,MAXTOL,JMAX,DAK)
      PARAM=OpenSeismoMatlab(DT,XGTT,'FS')
      PARAM=OpenSeismoMatlab(DT,XGTT,'BUTTERWORTHHIGH',BORDER,FLC)
      PARAM=OpenSeismoMatlab(DT,XGTT,'BUTTERWORTHLOW',BORDER,FHC)
      PARAM=OpenSeismoMatlab(DT,XGTT,'IDA',T,LAMBDAF,IM_DM,M,UY,PYSF,...
          KSI,ALGID,U0,UT0,RINF,MAXTOL,JMAX,DAK)
      Omit or set as empty ([]) the input arguments for which default
      values are desired.

  Description
      This function calculates the seismic parameters, develops various
      spectra and performs various analyses from an acceleration time
      history. More specifically, it calculates the following:
      1) Peak ground acceleration
      2) Peak ground velocity
      3) Peak ground displacement
      4) Total cumulative energy and normalized cumulative energy vs time
      5) Significant duration D_5_95 according to Trifunac & Brady (1975)
      6) Significant duration D_5_75
      7) Total Arias intensity (Ia)
      8) Velocity time history (with baseline correction or not)
      9) Displacement time history (with baseline correction or not)
      10) Resampled acceleration time history (i.e. the input acceleration
         time history with modified time step size)
      11) Linear elastic pseudo-acceleration response spectrum
      12) Linear elastic pseudo-velocity response spectrum
      13) Linear elastic displacement response spectrum
      14) Linear elastic velocity response spectrum
      15) Linear elastic acceleration response spectrum
      16) Constant ductility displacement response spectrum
      17) Constant ductility velocity response spectrum
      18) Constant ductility acceleration response spectrum
      19) Fourier amplitude spectrum
      20) Mean period (Tm)
      21) Lowpass Butterworth-filtered acceleration time history
      22) Highpass Butterworth-filtered acceleration time history
      23) Incremental Dynamic Analysis (IDA) of SDOF system excited with
          the input acceleration time history
```

```
    Depending on the value of SW, which determines the type of analysis
    that OpenSeismoMatlab performs, various additional parameters are
    needed as input by the user. All possible syntaxes appear above.

Input parameters
    DT [double(1 x 1)] is the size of the time step of the input
        acceleration time history xgtt.
    XGTT [double(:inf x 1)] is the input acceleration time history.
    SW [char(1 x :inf)] is a string which determines which parameters,
        spectras or analyses of the input acceleration time history will
        be calculated. SW can take one of the following values (strings
        are case insensitive):
        'TIMEHIST': the displacement, velocity and acceleration time
            histories are calculated.
        'RESAMPLE': the acceleration time history with modified time step
            size is calculated.
        'PGA': The peak ground acceleration is calculated.
        'PGV': The peak ground velocity is calculated.
        'PGD': The peak ground displacement is calculated.
        'ARIAS': The total cumulative energy, significant duration
            D_5_95 according to Trifunac & Brady (1975), significant
            duration D_5_75 and Arias intensity are calculated.
        'ES': The linear elastic response spectra and pseudospectra are
            calculated.
        'CDS': The constant ductility response spectra are calculated.
        'FS': The Fourier amplitude spectrum and the mean period are
            calculated.
        'BUTTERWORTHHIGH': The high-pass Butterworth filtered
            acceleration time history is calculated.
        'BUTTERWORTHLOW': The low-pass Butterworth filtered
            acceleration time history is calculated.
        'IDA': Incremental Dynamic Analysis of an elastoplastic SDOF
            system excited by the input acceleration time history is
            performed.
Additional required parameters
    BASELINESW [logical(1 x 1)] determines if baseline correction will be
        applied for the calculation of the various time histories.
    DTI [double(1 x 1)] is the new time step size for resampling of the
        input acceleration time history.
    T [double(:inf x 1)] contains the values of eigenperiods for
        which the response spectra are requested. Its length is the
        number of SDOF oscillators being analysed to produce the spectra.
        T must be a vector if SW='ES' or SW='CDS'. T must be scalar if
        SW='IDA'.
    KSI [double(1 x 1)] is the fraction of critical viscous damping.
    ALGID [char(1 x :inf)] is the algorithm to be used for the time
        integration, if applicable. It can be one of the following
        strings for superior optimally designed algorithms (strings are
        case sensitive):
            'generalized a-method': The generalized a-method (Chung &
            Hulbert, 1993)
            'HHT a-method': The Hilber-Hughes-Taylor method (Hilber,
            Hughes & Taylor, 1977)
            'WBZ': The Wood–Bossak–Zienkiewicz method (Wood, Bossak &
            Zienkiewicz, 1980)
            'U0-V0-Opt': Optimal numerical dissipation and dispersion
            zero order displacement zero order velocity algorithm
            'U0-V0-CA': Continuous acceleration (zero spurious root at
            the low frequency limit) zero order displacement zero order
            velocity algorithm
            'U0-V0-DA': Discontinuous acceleration (zero spurious root at
```

```
        the high frequency limit) zero order displacement zero order
        velocity algorithm
        'U0-V1-Opt': Optimal numerical dissipation and dispersion
        zero order displacement first order velocity algorithm
        'U0-V1-CA': Continuous acceleration (zero spurious root at
        the low frequency limit) zero order displacement first order
        velocity algorithm
        'U0-V1-DA': Discontinuous acceleration (zero spurious root at
        the high frequency limit) zero order displacement first order
        velocity algorithm
        'U1-V0-Opt': Optimal numerical dissipation and dispersion
        first order displacement zero order velocity algorithm
        'U1-V0-CA': Continuous acceleration (zero spurious root at
        the low frequency limit) first order displacement zero order
        velocity algorithm
        'U1-V0-DA': Discontinuous acceleration (zero spurious root at
        the high frequency limit) first order displacement zero order
        velocity algorithm
        'Newmark ACA': Newmark Average Constant Acceleration method
        'Newmark LA': Newmark Linear Acceleration method
        'Newmark BA': Newmark Backward Acceleration method
        'Fox-Goodwin': Fox-Goodwin formula
    Default value 'U0-V0-Opt'.
MU [double(1 x 1)] is the specified ductility for which the constant
    ductility response spectra are calculated.
PYSF [double(1 x 1)] is the post-yield stiffness factor, i.e. the
    ratio of the postyield stiffness to the initial stiffness. PYSF=0
    is not recommended for simulation of an elastoplastic system; a
    small positive value is always suggested due to numerical
    reasons. PYSF is ignored if MU=1. Default value 0.01.
BORDER [double(1 x 1)] is the order of the Butterworth filter that is
    applied for filtering of the acceleration time history.
FLC [double(1 x 1)] is the low cutoff frequency for the high-pass
    Butterworth filter.
FHC [double(1 x 1)] is the high cutoff frequency for the low-pass
    Butterworth filter.
LAMBDAF [double(:inf x 1)] contains the values of the scaling factor
    (lambda factor) for the incremental dynamic analysis.
IM_DM [char(1 x :inf)] is the Intensity Measure (IM) - Damage Measure
    (DM) pair that is to be calculated from the incremental dynamic
    analysis. IM_DM can take one of the following values (strings are
    case insensitive):
    'SA_MU': Spectral acceleration-ductility
    'PGD_MU': Peak displacement-ductility
    'PGV_MU': Peak velocity-ductility
    'PGA_MU': Peak acceleration-ductility
    'SA_DISP': Spectral acceleration-displacement
    'PGD_DISP': Peak displacement-displacement
    'PGV_DISP': Peak velocity-displacement
    'PGA_DISP': Peak acceleration-displacement
    'SA_VEL': Spectral acceleration-velocity
    'PGD_VEL': Peak displacement-velocity
    'PGV_VEL': Peak velocity-velocity
    'PGA_VEL': Peak acceleration-velocity
    'SA_ACC': Spectral acceleration-acceleration
    'PGD_ACC': Peak displacement-acceleration
    'PGV_ACC': Peak velocity-acceleration
    'PGA_ACC': Peak acceleration-acceleration
M [double(1 x 1)] is the mass of the SDOF oscillator.
UY [double(1 x 1)] is the yield displacement of the SDOF oscillator.
```

```
    U0 [double(1 x 1)] is the initial displacement of the SDOF
        oscillator.
    UT0 [double(1 x 1)] is the initial velocity of the SDOF oscillator.
    RINF [double(1 x 1)] is the minimum absolute value of the eigenvalues
        of the amplification matrix. For the amplification matrix see
        eq.(61) in Zhou & Tamma (2004). Default value 0.
    MAXTOL [double(1 x 1)] is the maximum tolerance of convergence of the
        Full Newton Raphson method for numerical computation of
        acceleration.
    JMAX [double(1 x 1)] is the maximum number of iterations per
        increment. If JMAX=0 then iterations are not performed and the
        MAXTOL parameter is not taken into account.
    DAK [double(1 x 1)] is the infinitesimal acceleration for the
        calculation of the derivetive required for the convergence of the
        Newton-Raphson iteration.
    DTTOL [double(1 x 1)] is the tolerance for resampling of the input
        acceleration time history. For a given eigenperiod T, resampling
        takes place if DT/T>dtTol. Default value 0.02.

Output parameters
    PARAM (structure) has the following fields:
        PARAM.time [double(:inf x 1)] Time
        PARAM.acc [double(:inf x 1)] Acceleration time history
        PARAM.vel [double(:inf x 1)] Velocity time history
        PARAM.disp [double(:inf x 1)] Displacement time history
        PARAM.PGA [double(1 x 1)] Peak ground acceleration
        PARAM.PGV [double(1 x 1)] Peak ground velocity
        PARAM.PGD [double(1 x 1)] Peak ground displacement
        PARAM.Ecum [double(1 x 1)] Total cumulative energy
        PARAM.EcumTH [double(:inf x 1)] normalized cumulative
            energy vs time
        PARAM.t_5_95 [double(1 x 2)] Time instants at which 5% and 95% of
            cumulative energy have occurred
        PARAM.Td_5_95 [double(1 x 1)] Time between when 5% and 95% of
            cumulative energy has occurred (significant duration
            according to Trifunac-Brady (1975))
        PARAM.t_5_75 [double(1 x 2)] Time instants at which 5% and 75% of
            cumulative energy have occurred
        PARAM.Td_5_75 [double(1 x 1)] Time between when 5% and 75% of
            cumulative energy has occurred
        PARAM.arias [double(1 x 1)] Total Arias intensity (Ia)
        PARAM.PSa [double(:inf x 1)] Linear elastic pseudo-acceleration
            response spectrum
        PARAM.PSv [double(:inf x 1)] Linear elastic pseudo-velocity
            response spectrum
        PARAM.Sd [double(:inf x 1)] Linear elastic displacement response
            spectrum
        PARAM.Sv [double(:inf x 1)] Linear elastic velocity response
            spectrum
        PARAM.Sa [double(:inf x 1)] Linear elastic acceleration response
            spectrum
        PARAM.SievABS [double(:inf x 1)] Linear elastic absolute input
            energy equivalent velocity spectrum
        PARAM.SievREL [double(:inf x 1)] Linear elastic relative input
            energy equivalent velocity spectrum
        PARAM.PredPSa [double(1 x 1)] Predominant acceleration of the PSa
            spectrum
        PARAM.PredPeriod [double(1 x 1)] Predominant period of the PSa
            spectrum
        PARAM.CDPSa [double(:inf x 1)] Constant ductility
            pseudo-acceleration response spectrum
```

```
        PARAM.CDPSv [double(:inf x 1)] Constant ductility pseudo-velocity
            response spectrum
        PARAM.CDSd [double(:inf x 1)] Constant ductility displacement
            response spectrum
        PARAM.CDSv [double(:inf x 1)] Constant ductility velocity
            response spectrum
        PARAM.CDSa [double(:inf x 1)] Constant ductility acceleration
            response spectrum
        PARAM.fyK [double(:inf x 1)] yield limit that each SDOF must have
            in order to attain ductility equal to PARAM.muK.
        PARAM.muK [double(:inf x 1)] achieved ductility for each period
            (each SDOF).
        PARAM.iterK [double(:inf x 1)] number of iterations needed for
            convergence for each period (each SDOF).
        PARAM.FAS [double(1:2^(nextpow2(length(xgtt))-1) x 1)] Fourier
            amplitude spectrum
        PARAM.Tm [double(1 x 1)] Mean period (Tm)
        PARAM.Fm [double(1 x 1)] Mean frequency (Fm)
        param.DM [double(:inf x 1)] is the damage measure (DM) of the
            incremental dynamic analysis
        param.IM [double(:inf x 1)] is the intensity measure (IM) of the
            incremental dynamic analysis
```

```
Copyright (c) 2018-2022
    George Papazafeiropoulos
    Major, Infrastructure Engineer, Hellenic Air Force
    Civil Engineer, M.Sc., Ph.D.
    Email: gpapazafeiropoulos@yahoo.gr
```

Published with MATLAB® R2017b

baselineCorr

Apply baseline correction in OpenSeismoMatlab

Contents

Earthquake motion

Load earthquake data

```
fid=fopen('Imperial_Valley_El_Centro_9_EW.dat','r');
text=textscan(fid,'%f %f');
fclose(fid);
time=text{1,1};
xgtt1=text{1,2};
dt=time(2)-time(1);
```

Integrate without baseline correction

Calculate the velocity and displacement time histories

```
xgt1 = cumtrapz(time,xgtt1);
xg1 = cumtrapz(time,xgt1);
```

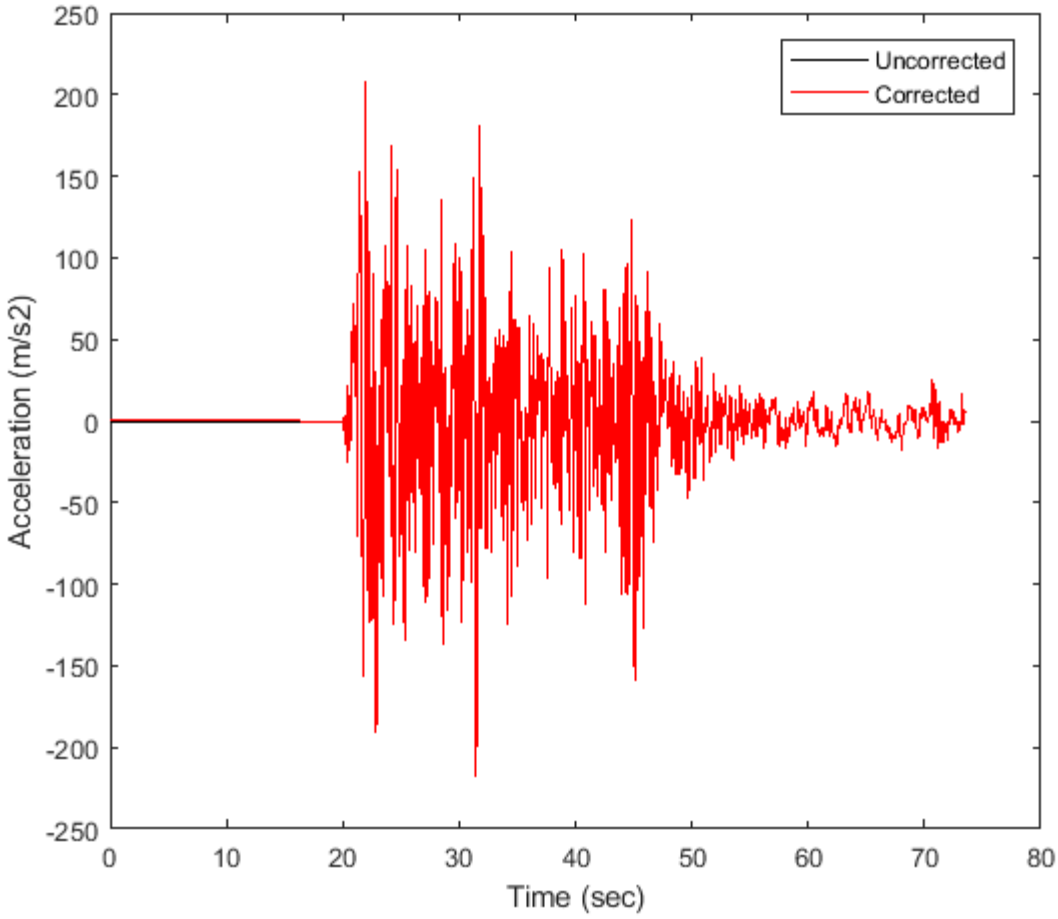
Integrate with baseline correction

Calculate the displacement, velocity and acceleration time histories

```
[xg2, xgt2, xgtt2] = baselineCorr(time,xgtt1);
```

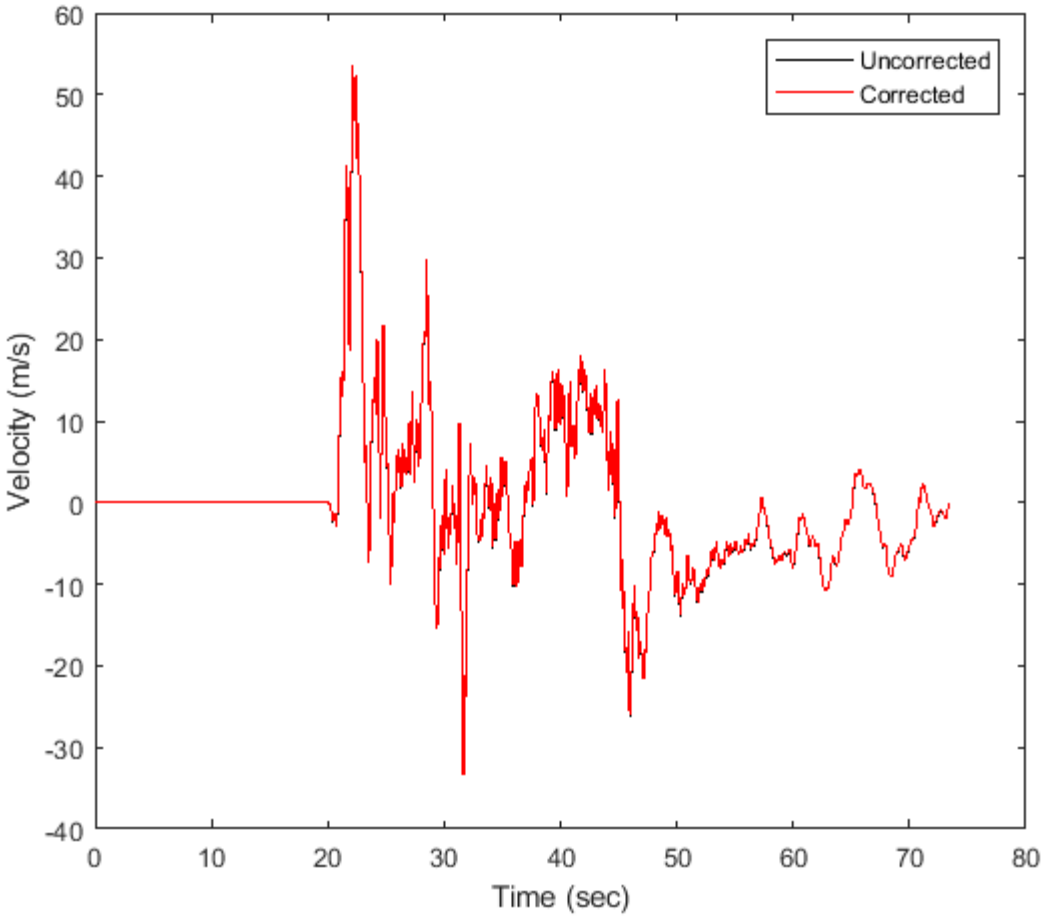
Plot corrected and uncorrected acceleration

```
figure()
plot(time,xgtt1,'k','LineWidth',1)
hold on
plot(time,xgtt2,'r','LineWidth',1)
hold off
ylabel('Acceleration (m/s2)')
xlabel('Time (sec)')
legend('Uncorrected','Corrected')
```



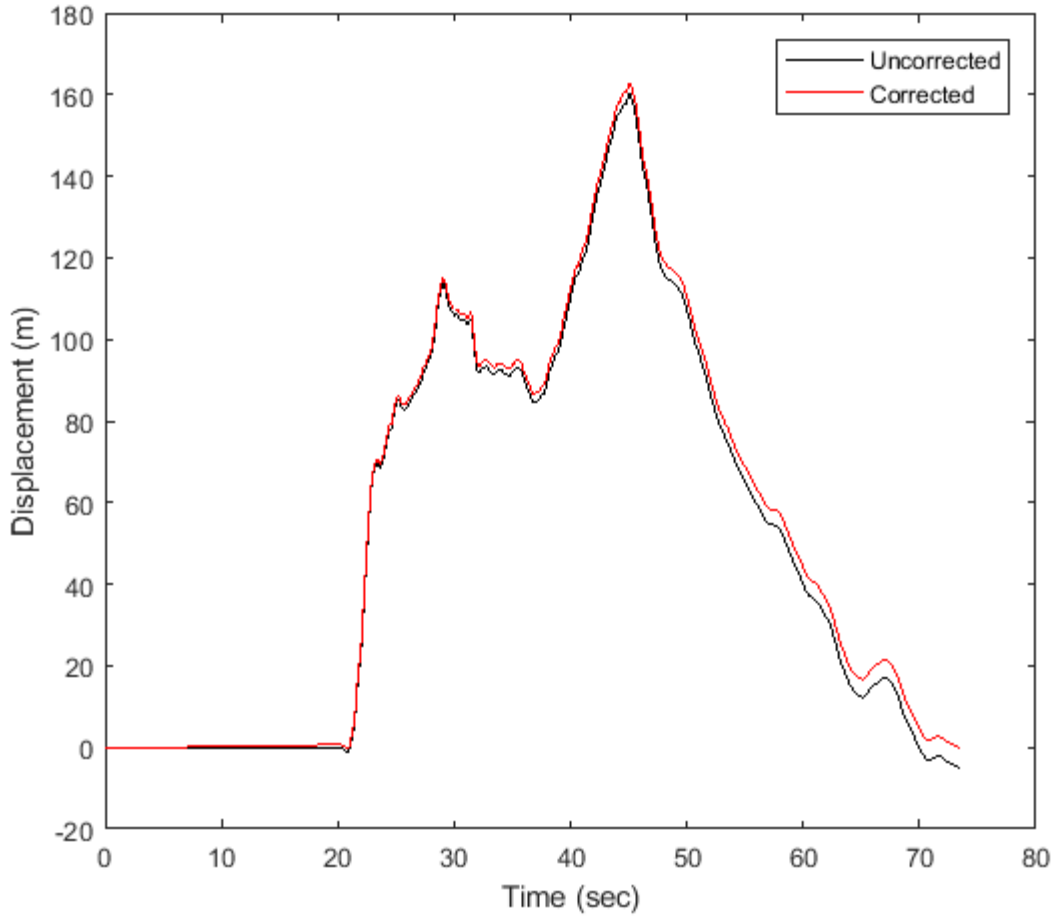
Plot corrected and uncorrected velocity

```
figure()
plot(time,xgt1,'k','LineWidth',1)
hold on
plot(time,xgt2,'r','LineWidth',1)
hold off
ylabel('Velocity (m/s)')
xlabel('Time (sec)')
legend('Uncorrected','Corrected')
```



Plot corrected and uncorrected displacement

```
figure()
plot(time,xg1,'k','LineWidth',1)
hold on
plot(time,xg2,'r','LineWidth',1)
hold off
ylabel('Displacement (m)')
xlabel('Time (sec)')
legend('Uncorrected','Corrected')
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

CDReSp

Calculate constant ductility response spectra in OpenSeismoMatlab

Contents

Earthquake motion

For reproducibility

```
rng(0)
```

Generate earthquake data

```
dt=0.02;
N=10;
a=rand(N,1)-0.5;
b=100*pi*rand(N,1);
c=pi*(rand(N,1)-0.5);
t=(0:dt:(100*dt))';
xgtt=zeros(size(t));
for i=1:N
    xgtt=xgtt+a(i)*sin(b(i)*t+c(i));
end
```

Setup parameters for CDReSp function

Eigenperiods

```
T=(0.04:0.04:4)';
```

Critical damping ratio

```
ksi=0.05;
```

Ductility

```
mu=2;
```

Maximum number of iterations

```
n=50;
```

Tolerance for convergence to target ductility

```
tol=0.01;
```

Post-yield stiffness factor

```
pysf=0.1;
```

Maximum ratio of the integration time step to the eigenperiod

```
dtTol=0.02;
```

Algorithm to be used for the time integration

```
AlgID='U0-V0-Opt';
```

Minimum absolute value of the eigenvalues of the amplification matrix

```
rinf=1;
```

Maximum tolerance of convergence for time integration algorithm

```
maxtol=0.01;
```

Maximum number of iterations per integration time step

```
jmax=200;
```

Infinitesimal acceleration

```
dak=eps;
```

Calculate spectra and pseudospectra

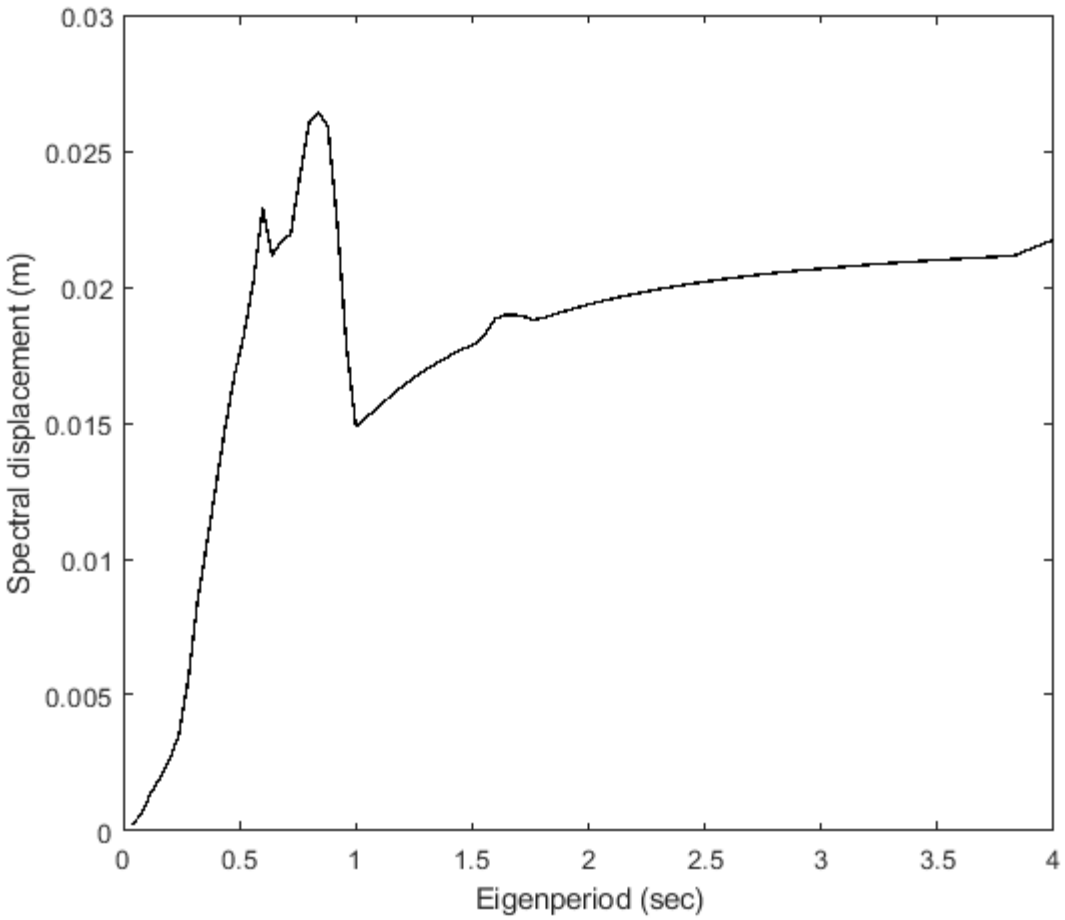
Apply CDReSp

```
[CDPSa,CDPSv,CDSd,CDSv,CDSa,fyK,muK,iterK]=CDReSp(dt,xgtt,T,ksi,...
    mu,n,tol,pysf,dtTol,AlgID,rinf,maxtol,jmax,dak);
```

Plot the spectra and pseudospectra

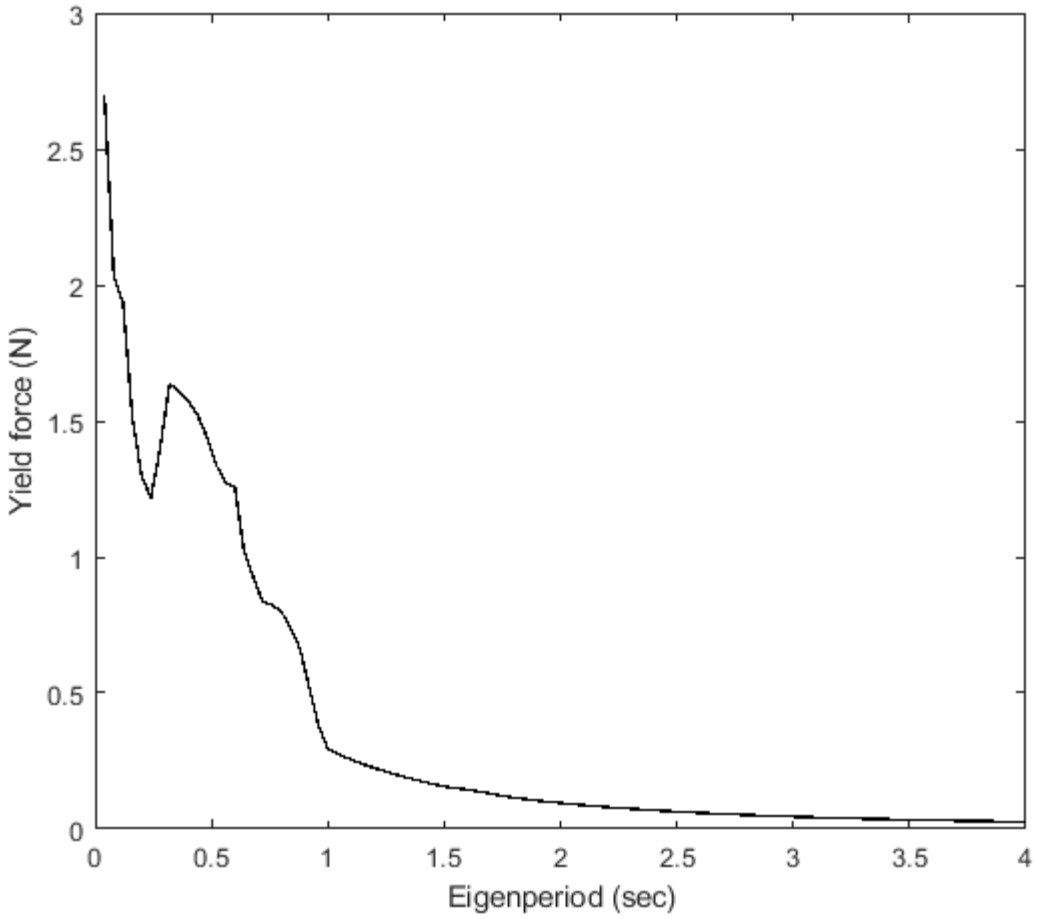
Constant ductility spectral displacement spectrum

```
figure()
plot(T,CDSd,'k','LineWidth',1)
ylabel('Spectral displacement (m)')
xlabel('Eigenperiod (sec)')
```



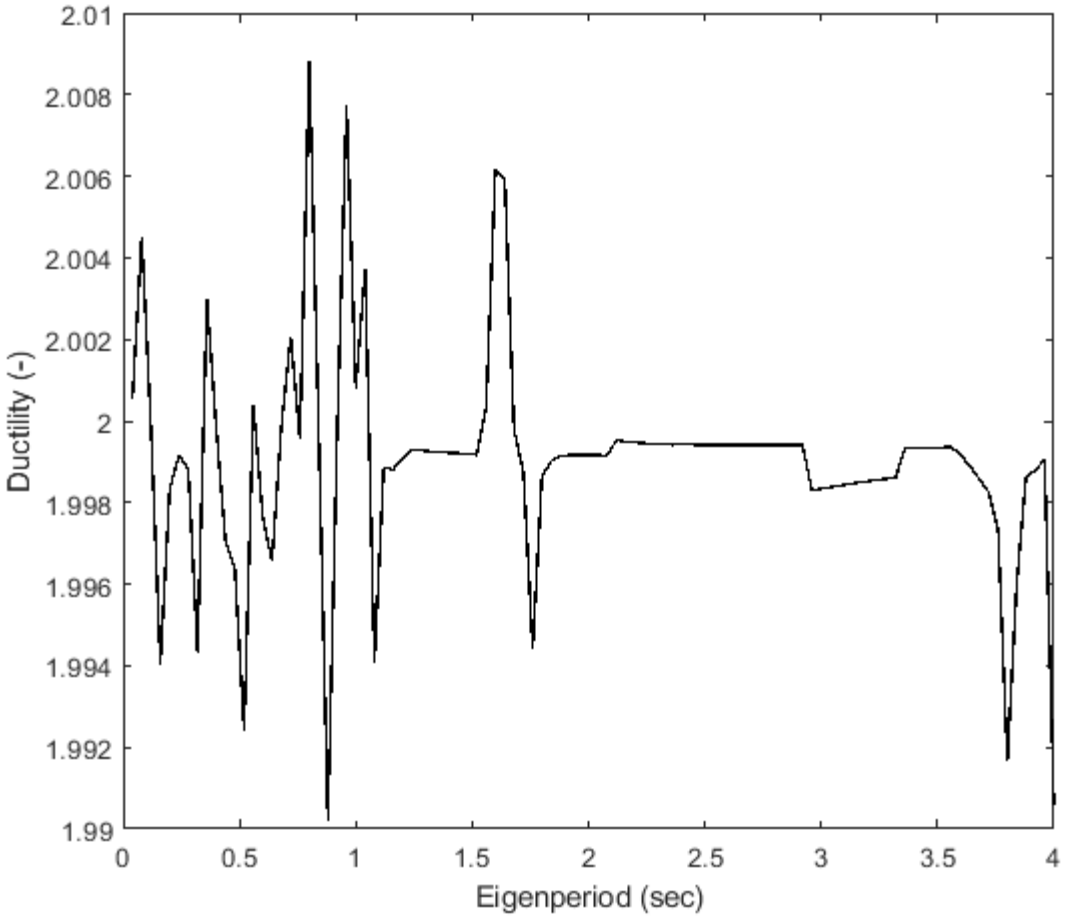
Constant ductility yield force spectrum

```
figure()
plot(T,fyK,'k','LineWidth',1)
ylabel('Yield force (N)')
xlabel('Eigenperiod (sec)')
```



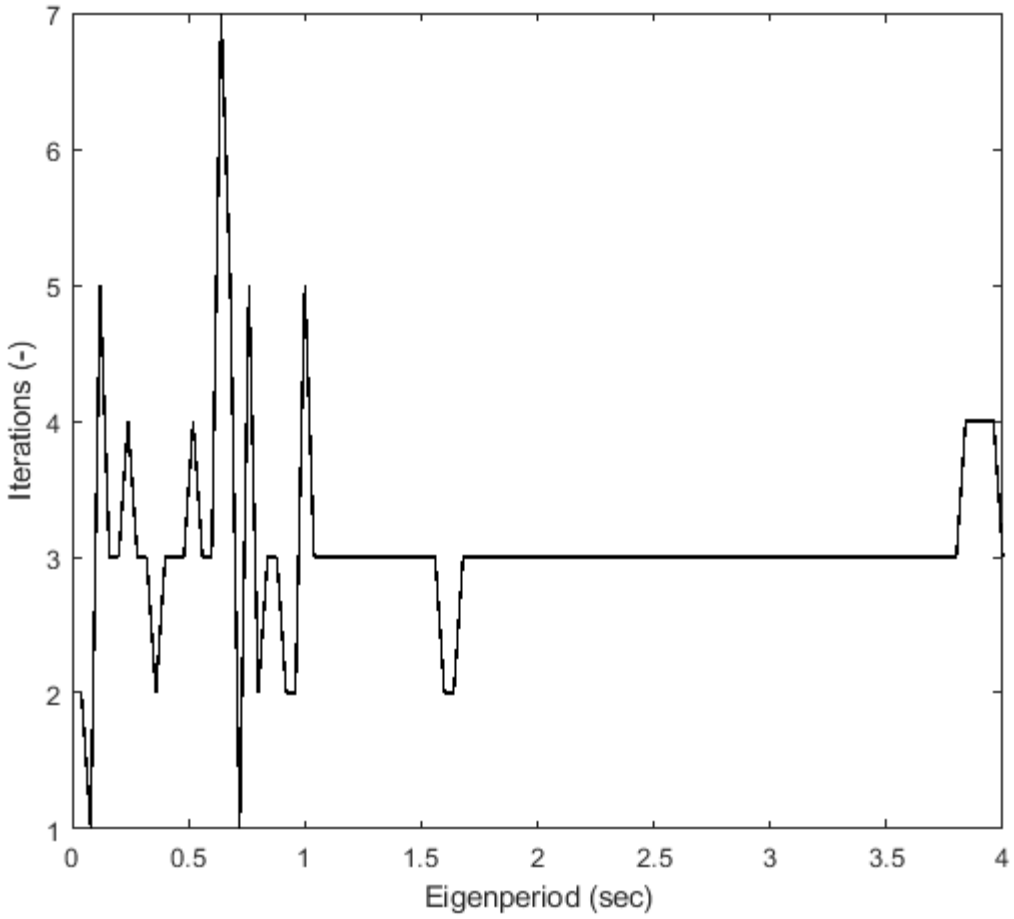
Achieved ductility

```
figure()
plot(T,muK,'k','LineWidth',1)
ylabel('Ductility (-)')
xlabel('Eigenperiod (sec)')
```



Iterations

```
figure()
plot(T,iterK,'k','LineWidth',1)
ylabel('Iterations (-)')
xlabel('Eigenperiod (sec)')
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Constant ductility response spectra

Generate the constant ductility response spectra and associated results of an earthquake suite using OpenSeismoMatlab.

Contents

Input

earthquake motions

```
eqmotions={'Imperial Valley'; % Imperial valley 1979
    'Kocaeli';
    'Loma Prieta';
    'Northridge';
    'San Fernando';
    'Spitak';
    'Cape Mendocino';
    'ChiChi';
    'El Centro'; % Imperial valley 1940
    'Hollister';
    'Kobe'};
```

Set the eigenperiod range for which the response spectra will be calculated.

```
Tspectra=(0.08:0.08:4)';
```

Set critical damping ratio of the response spectra to be calculated.

```
ksi=0.05;
```

Set the target ductility (not used here)

```
mu=2;
```

Set the postyield stiffness factor for each earthquake

```
p=[0.02;0.01;0.02;0.01;0.01;0.01;0.02;0.01;0.01;0.01;0.01];
```

Extract nonlinear response spectra

```
sw='cds';
```

Calculation

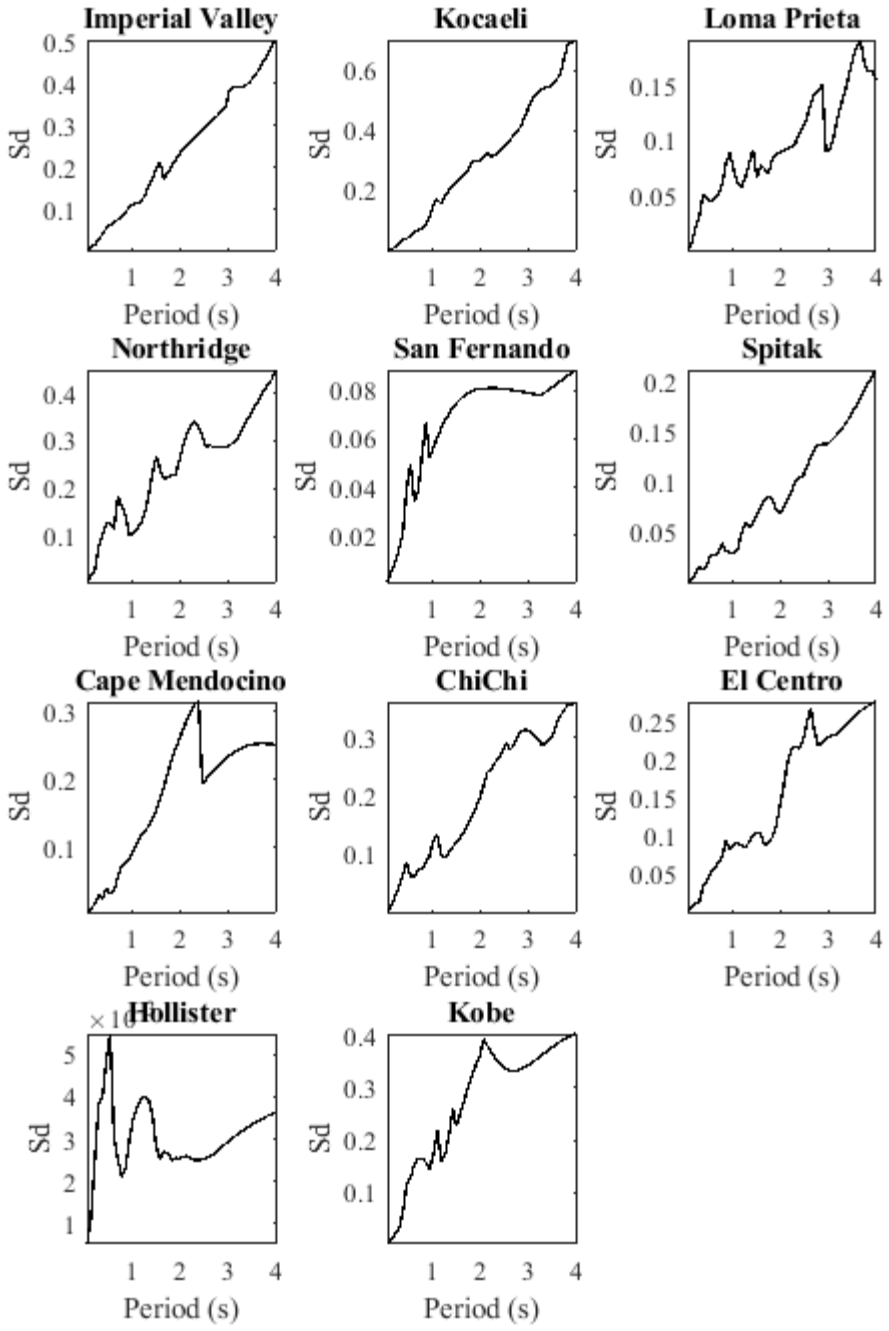
Initialize CDRS

```
CDRS=cell(numel(eqmotions),1);
% Calculation of peak values
for i=1:numel(eqmotions)
    % earthquake
    data=load([eqmotions{i},'.dat']);
    t=data(:,1);
    dt=t(2)-t(1);
    xgtt=data(:,2);
    S=OpenSeismoMatlab(dt,xgtt,sw,Tspectra,ksi,mu,p(i));
    CDRS{i}=[S.Period,S.CDSd,S.CDSv,S.CDPSa,S.fyK,S.muK,S.iterK];
end
```

Output

Plot constant ductility spectral displacement

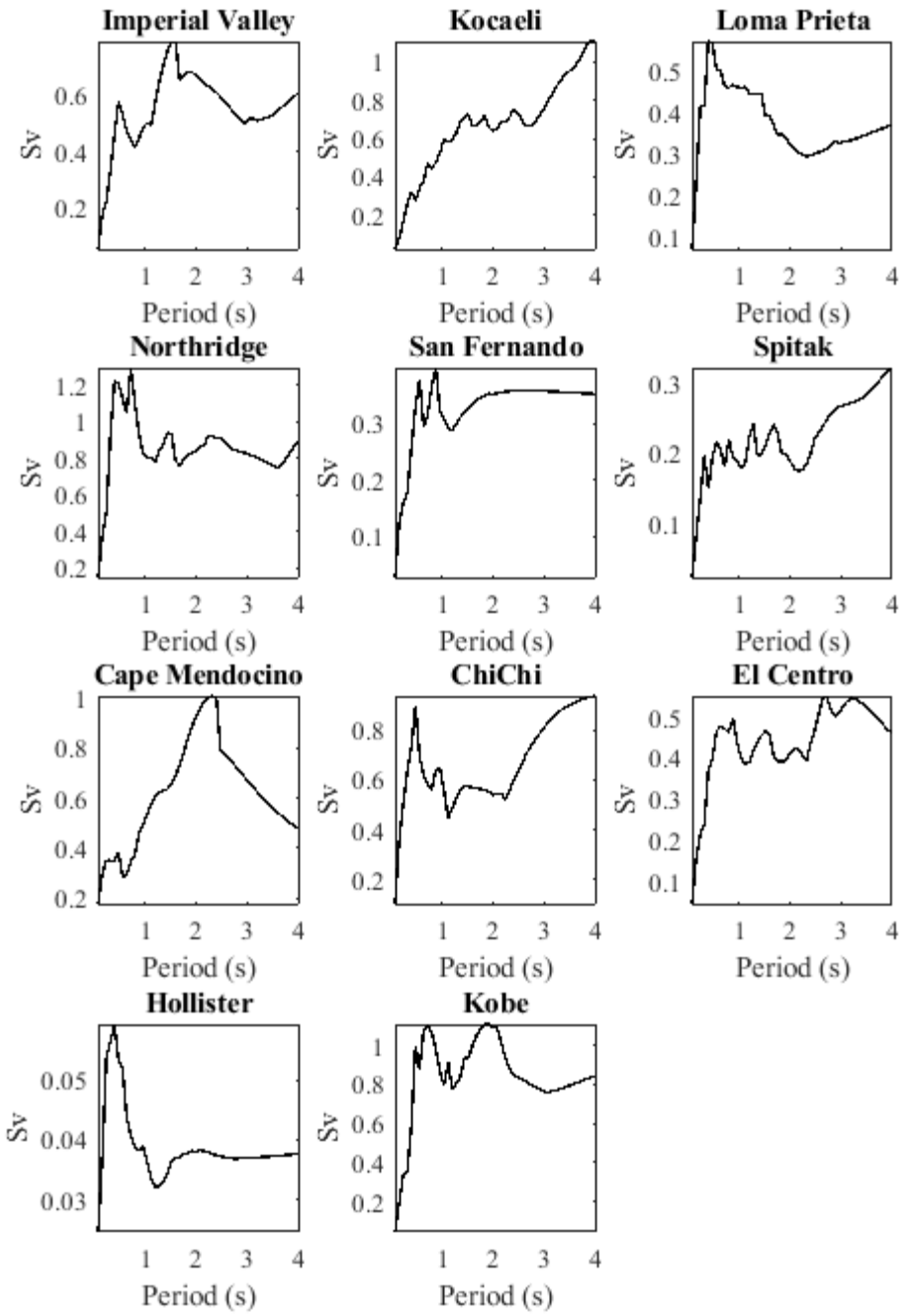
```
Fig1 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(CDRS{i}(:,1),CDRS{i}(:,2),'k','LineWidth',1);
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('Sd','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
    axis tight
end
```



Plot constant ductility spectral velocity

```
Fig2 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(CDRS{i}(:,1),CDRS{i}(:,3),'k','LineWidth',1);
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('Sv','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
```

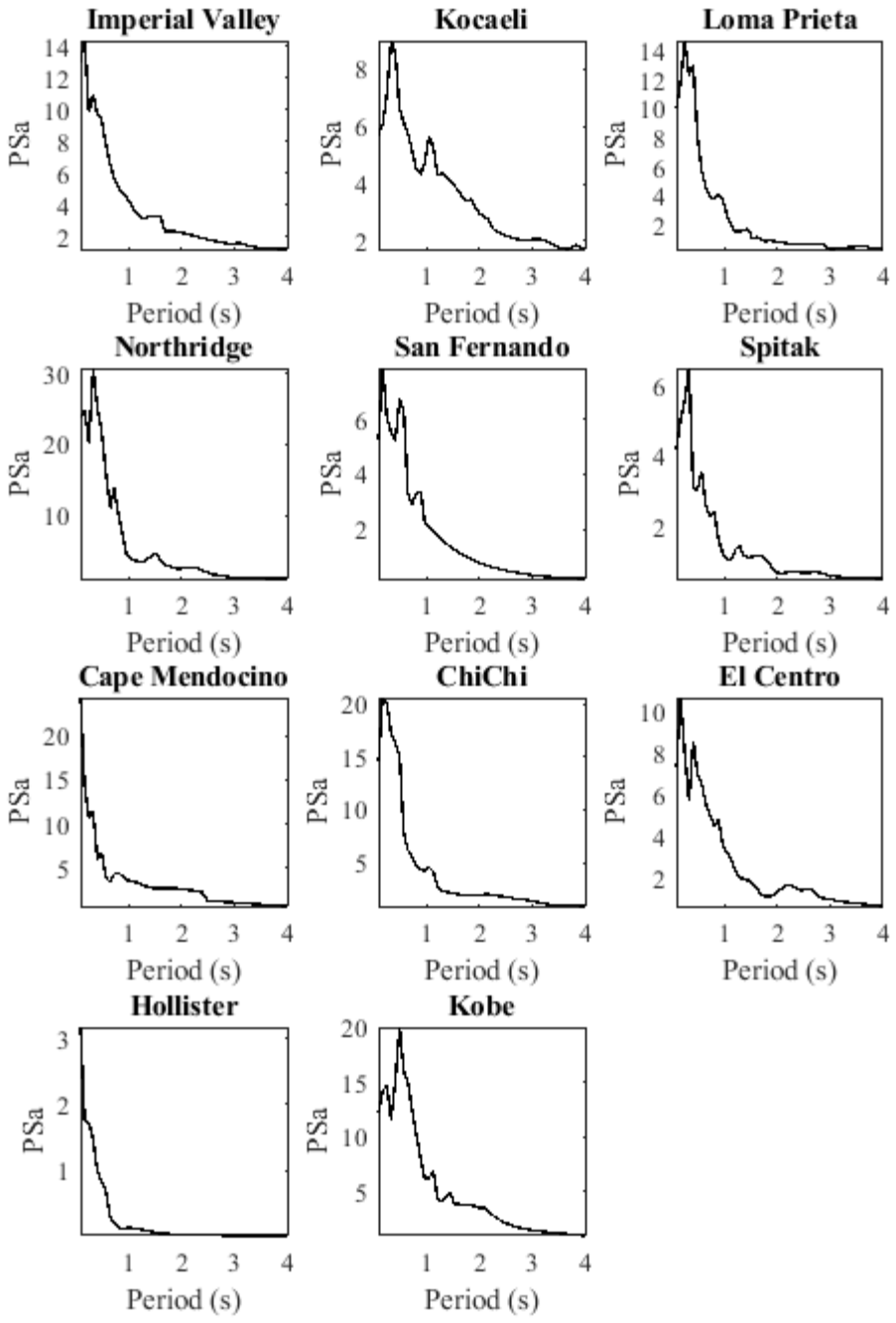
```
    axis tight
end
```



Plot constant ductility spectral acceleration

```
Fig3 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(CDRS{i}(:,1),CDRS{i}(:,4),'k','LineWidth',1);
```

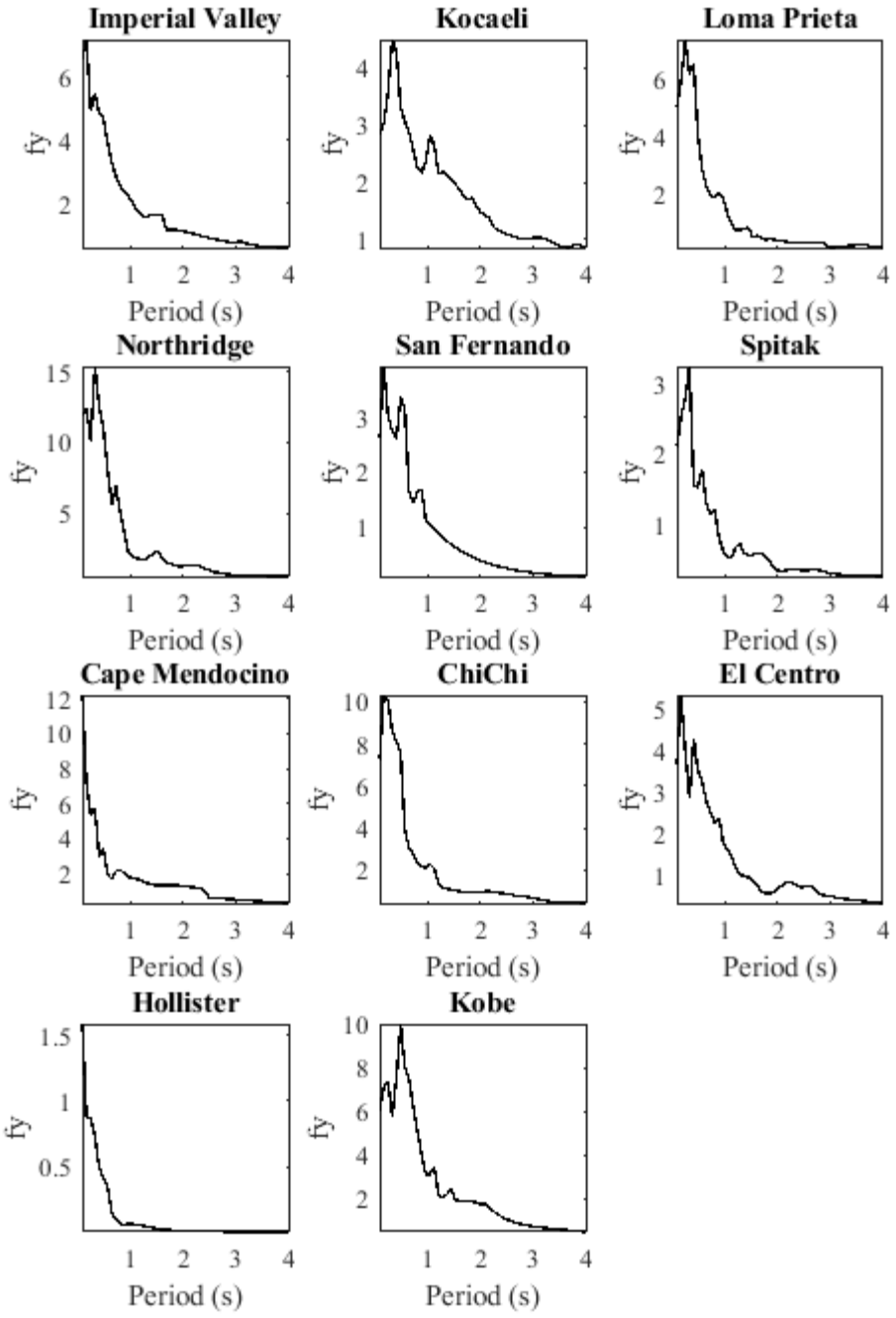
```
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('PSa','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
    axis tight
end
```



Plot constant ductility spectral yield limit

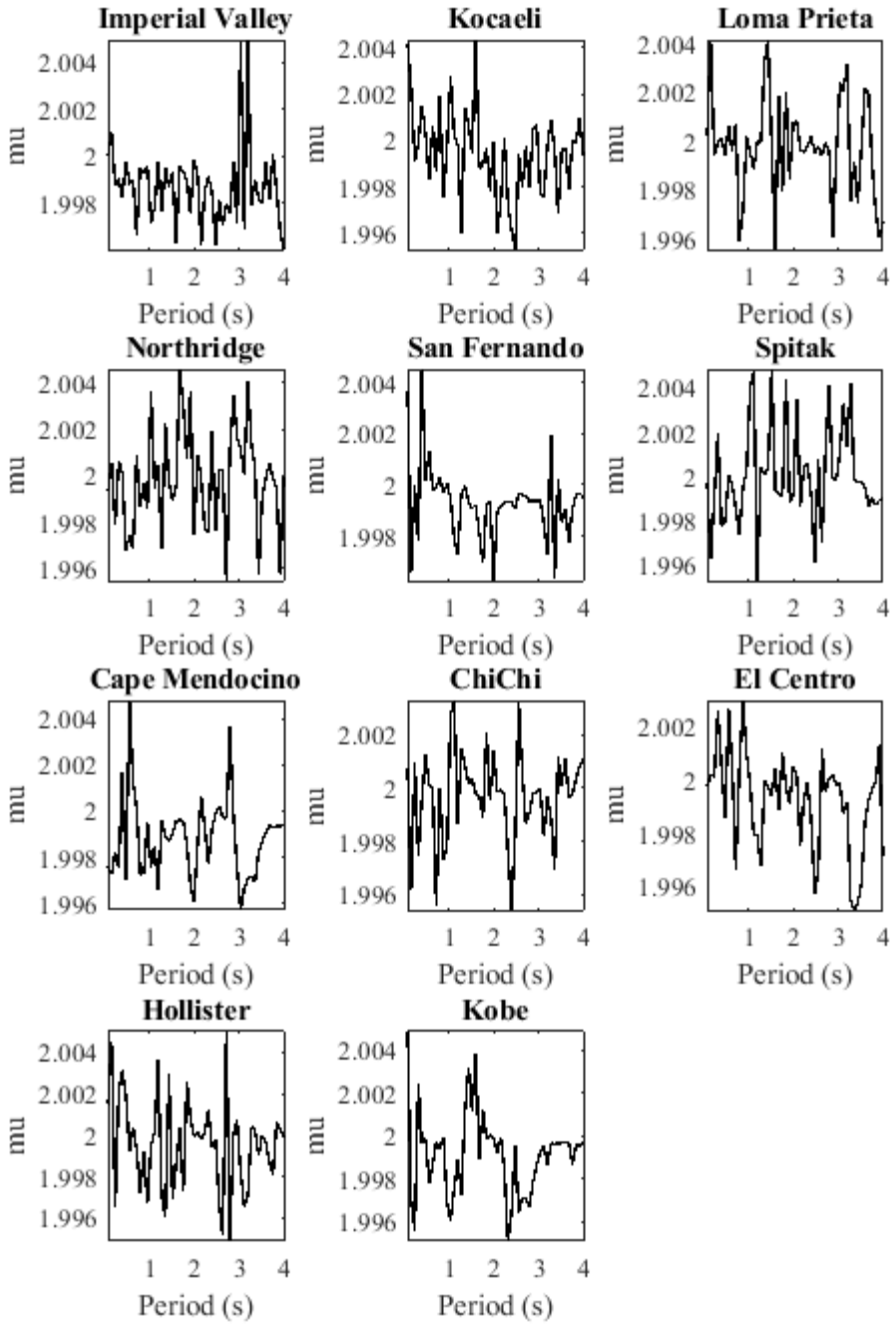
```
Fig4 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
```

```
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(CDRS{i}(:,1),CDRS{i}(:,5),'k','LineWidth',1);
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('fy','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
    axis tight
end
```



Plot constant ductility spectral achieved ductility

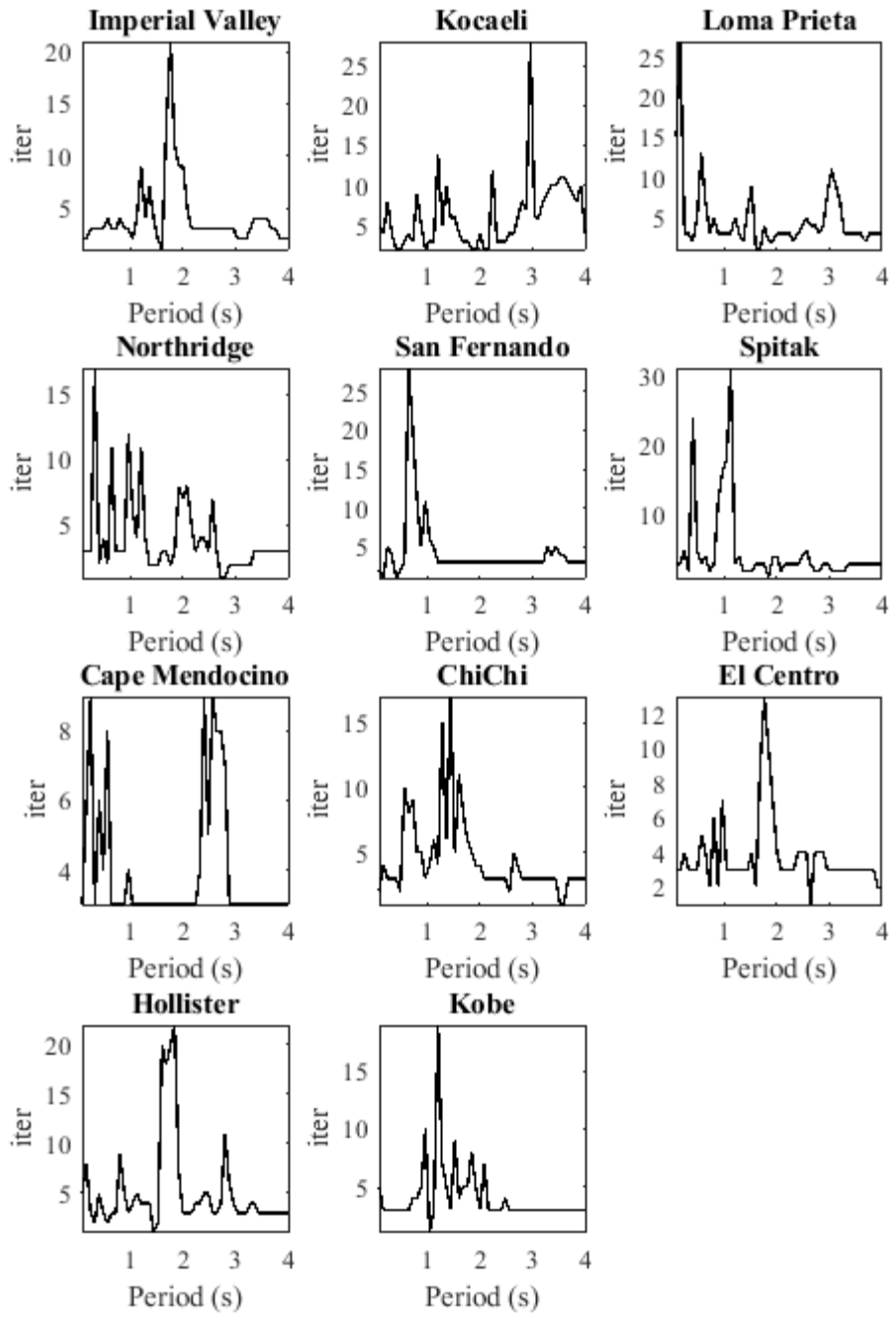
```
Fig5 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(CDRS{i}(:,1),CDRS{i}(:,6),'k','LineWidth',1);
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('mu','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
    axis tight
end
```



Plot constant ductility spectral number of iterations needed for convergence

```
Fig6 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(CDRS{i}(:,1),CDRS{i}(:,7),'k','LineWidth',1);
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('iter','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
```

```
    axis tight
end
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Fourier spectra

Generate the Fourier spectra of an earthquake suite using OpenSeismoMatlab

Contents

Input

earthquake motions

```
eqmotions={'Imperial Valley'; % Imperial valley 1979
    'Kocaeli';
    'Loma Prieta';
    'Northridge';
    'San Fernando';
    'Spitak';
    'Cape Mendocino';
    'ChiChi';
    'El Centro'; % Imperial valley 1940
    'Hollister';
    'Kobe'};
```

Switch

```
sw='fs';
```

Extract fourier spectra

Initialize cell of Fourier spectra

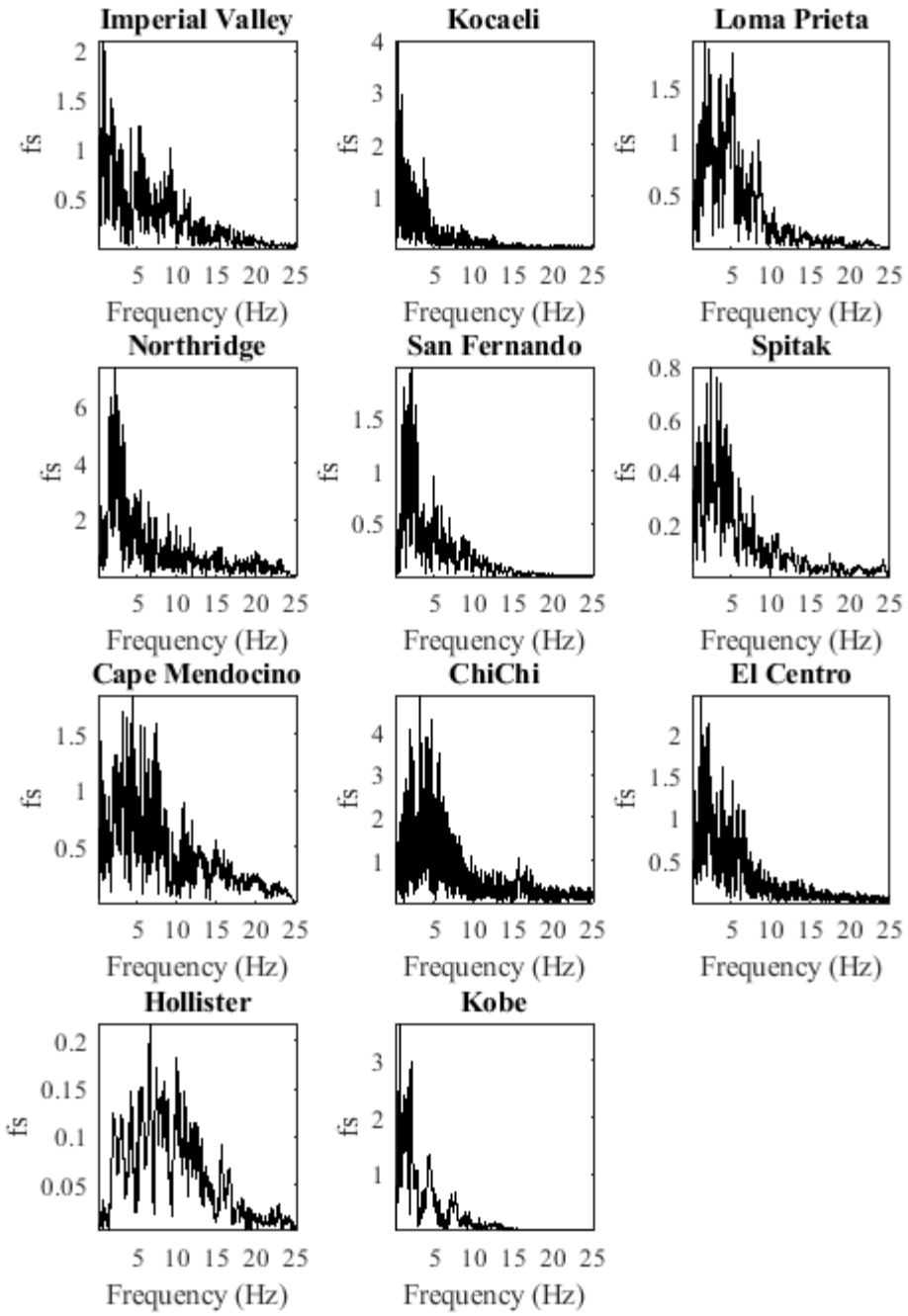
```
Fourier=cell(numel(eqmotions),1);
% Calculation of Fourier spectra
for i=1:numel(eqmotions)
    % earthquake
    data=load([eqmotions{i},'.dat']);
    t=data(:,1);
    dt=t(2)-t(1);
    xgtt=data(:,2);
    S=OpenSeismoMatlab(dt,xgtt,sw);
    Fourier{i}=[S.freq,S.FAS];
end
```

Output

Plot Fourier amplitude

```
```

```
Fig1 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(Fourier{i}(:,1),Fourier{i}(:,2),'k','LineWidth',1);
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('fs','FontName','Times New Roman')
    xlabel('Frequency (Hz)','FontName','Times New Roman')
    axis tight
end
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

General demonstration of the OpenSeismoMatlab capabilities

This is to demonstrate the capabilities of OpenSeismoMatlab and also to verify that OpenSeismoMatlab works properly for all of its possible options and types of application, and yields meaningful results. **All** capabilities of OpenSeismoMatlab are tested in this example

Contents

Load earthquake data

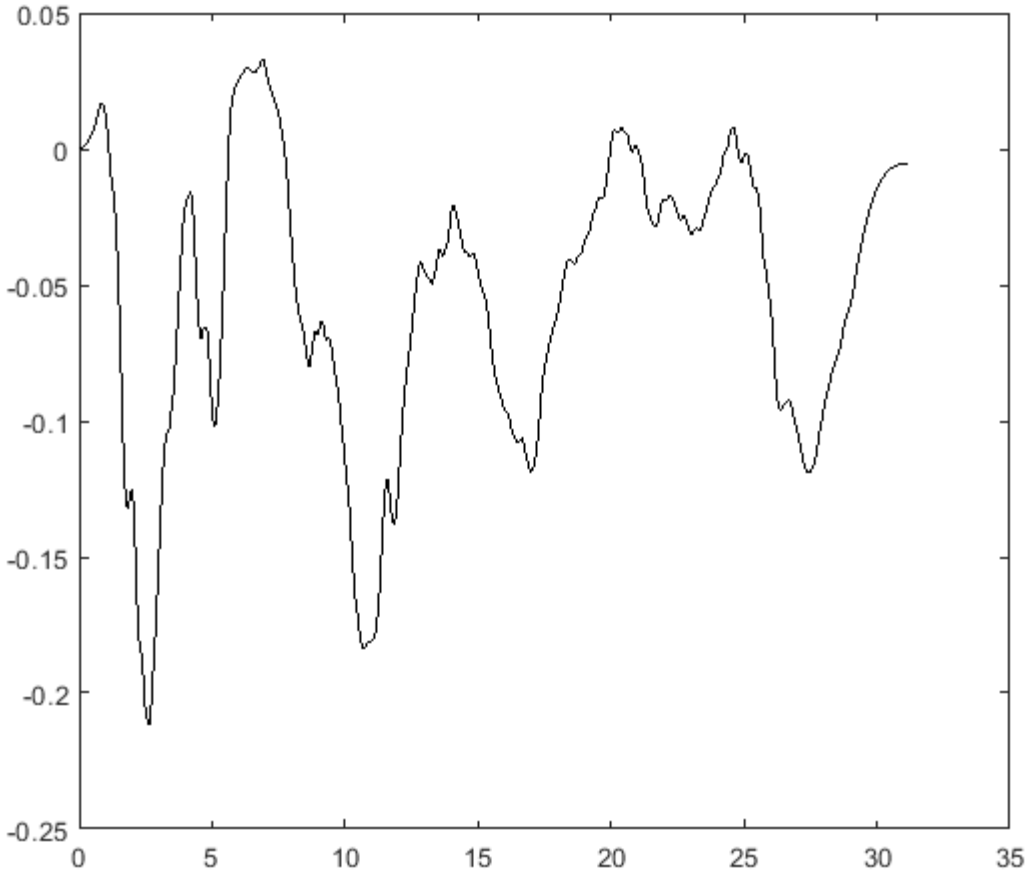
Earthquake acceleration time history of the El Centro earthquake will be used (El Centro, 1940, El Centro Terminal Substation Building)

```
fid=fopen('elcentro.dat','r');
text=textscan(fid,'%f %f');
fclose(fid);
t=text{1,1};
dt=t(2)-t(1);
xgtt=text{1,2};
```

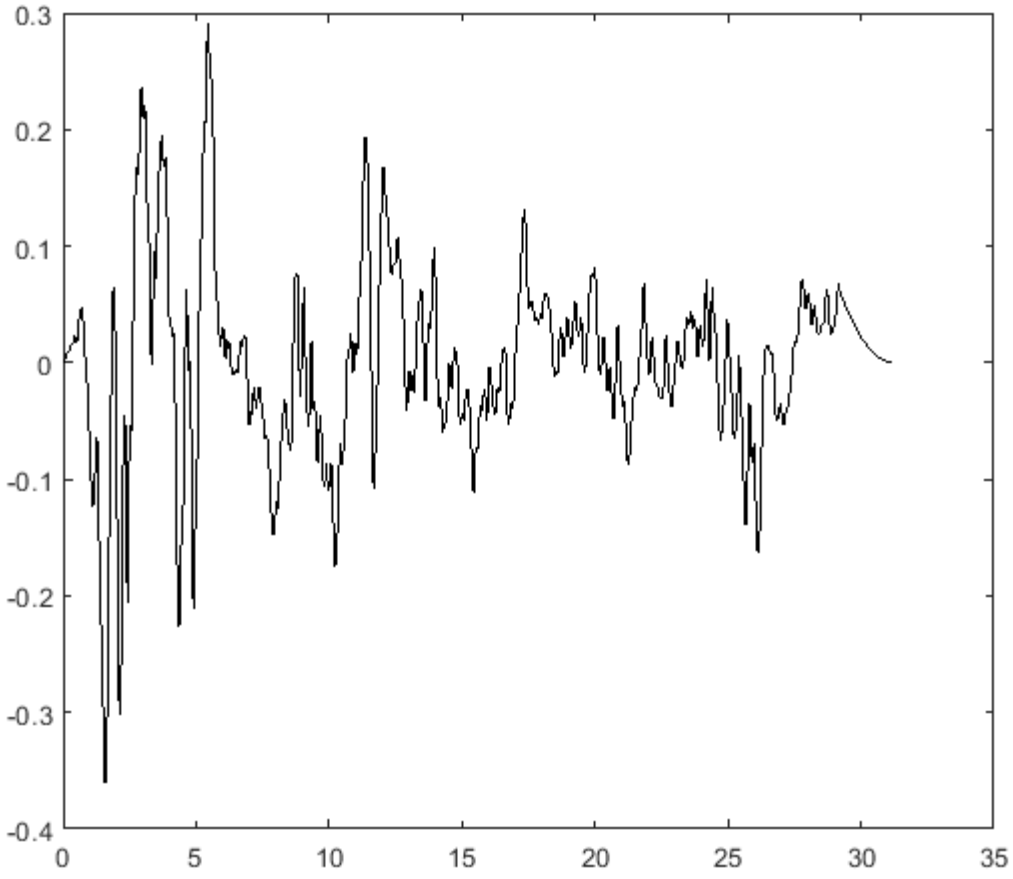
Time histories without baseline correction

```
sw='timehist';
baselineSw=false;
S1=OpenSeismoMatlab(dt,xgtt,sw,baselineSw);
```

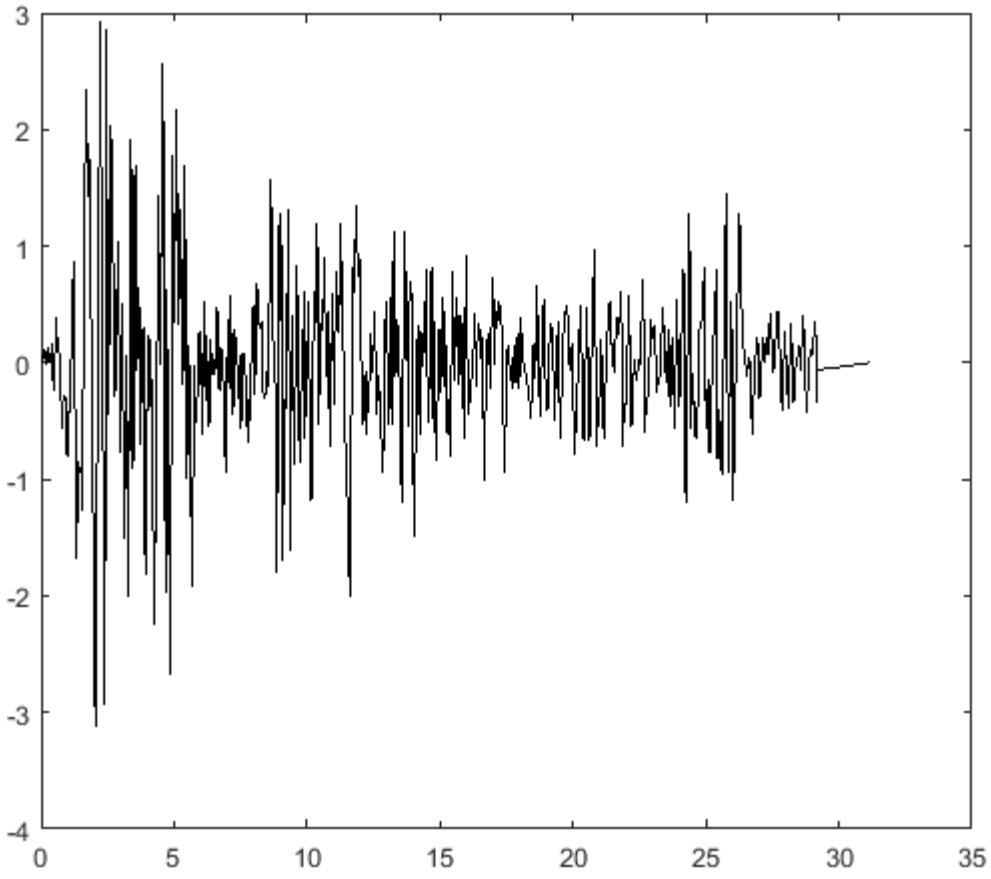
```
figure(1)
plot(S1.time,S1.disp,'k','LineWidth',1)
```




```
figure(2)
plot(S1.time,S1.vel,'k','LineWidth',1)
```



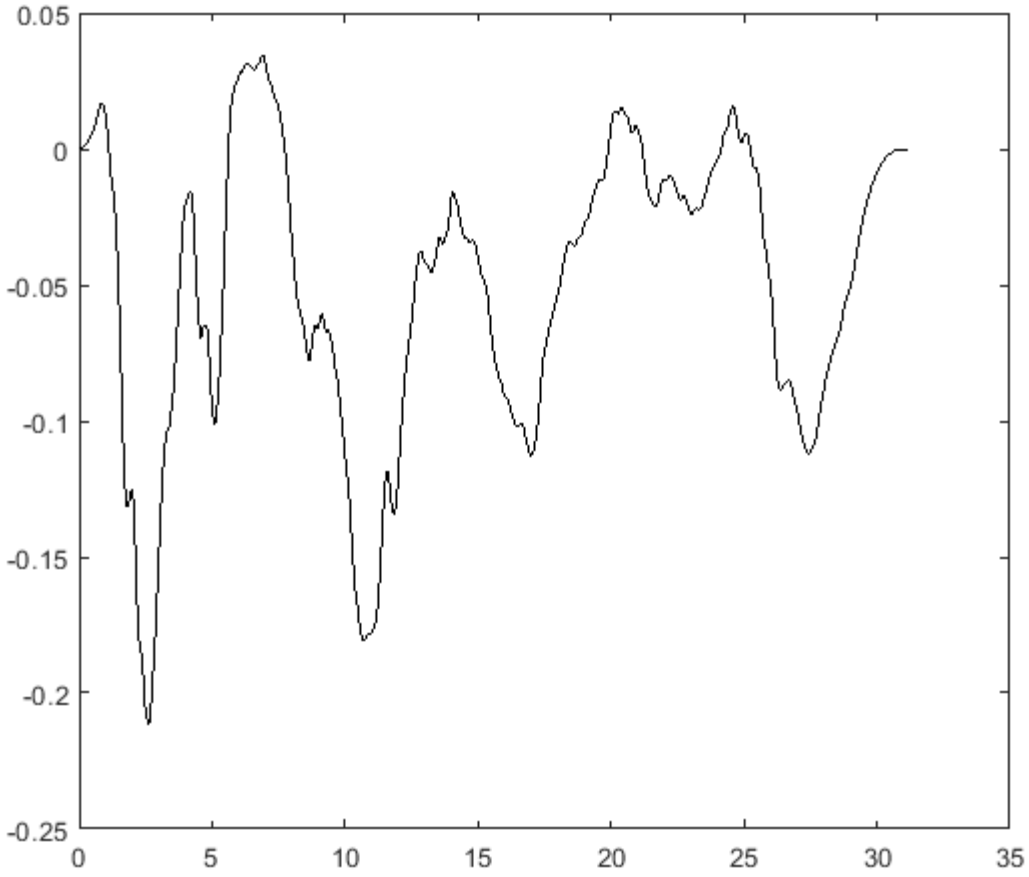

```
figure(3)
plot(S1.time,S1.acc,'k','LineWidth',1)
```



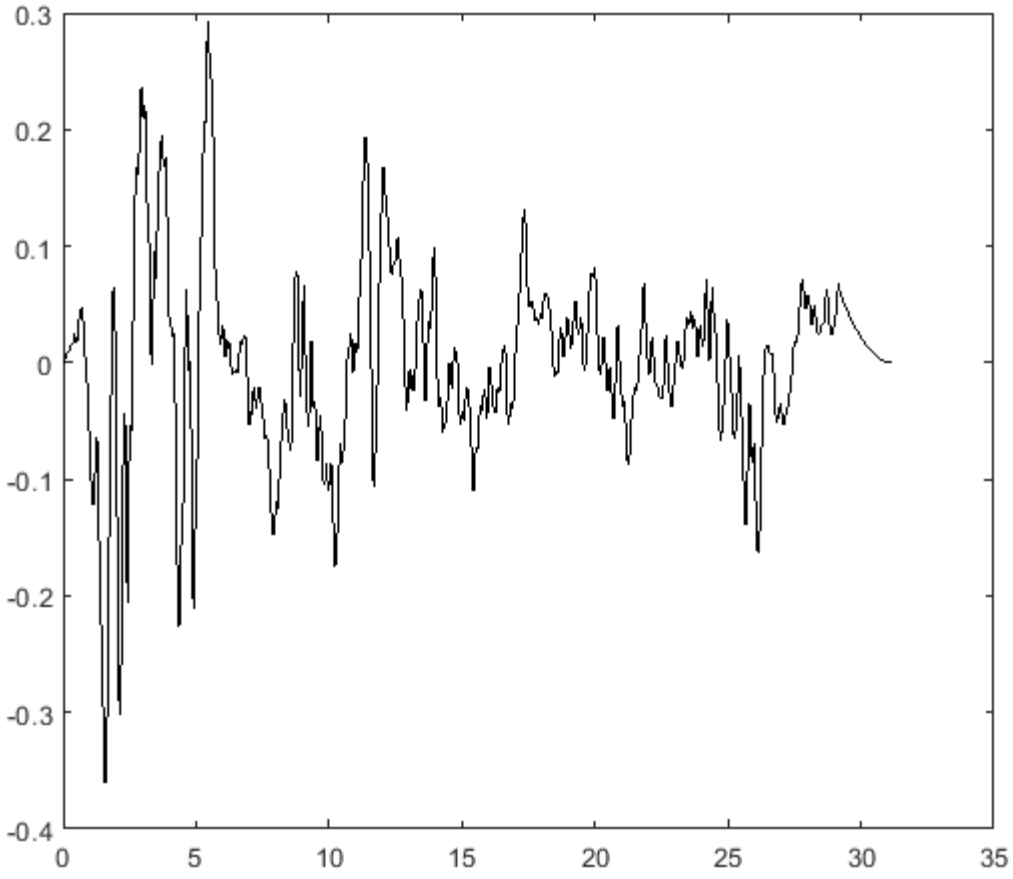
Time histories with baseline correction

```
sw='timehist';
baselineSw=true;
S2=OpenSeismoMatlab(dt,xgtt,sw,baselineSw);
```

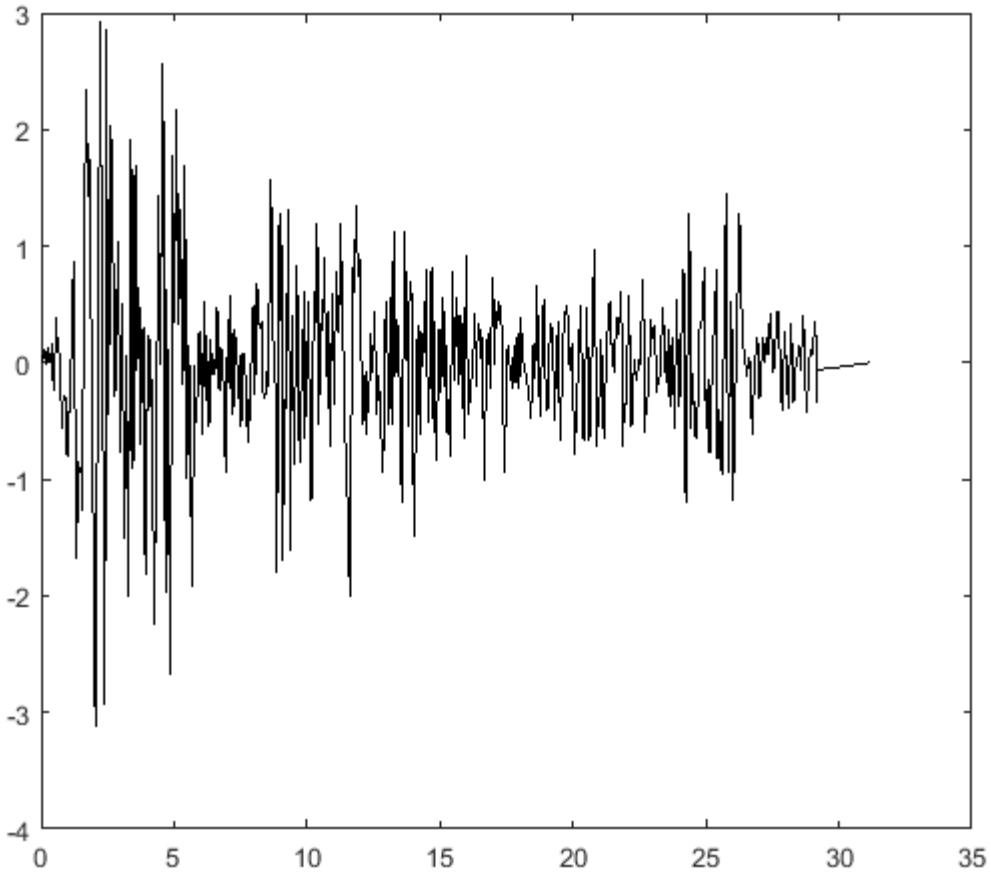
```
figure(4)
plot(S2.time,S2.disp,'k','LineWidth',1)
```




```
figure(5)
plot(S2.time,S2.vel,'k','LineWidth',1)
```



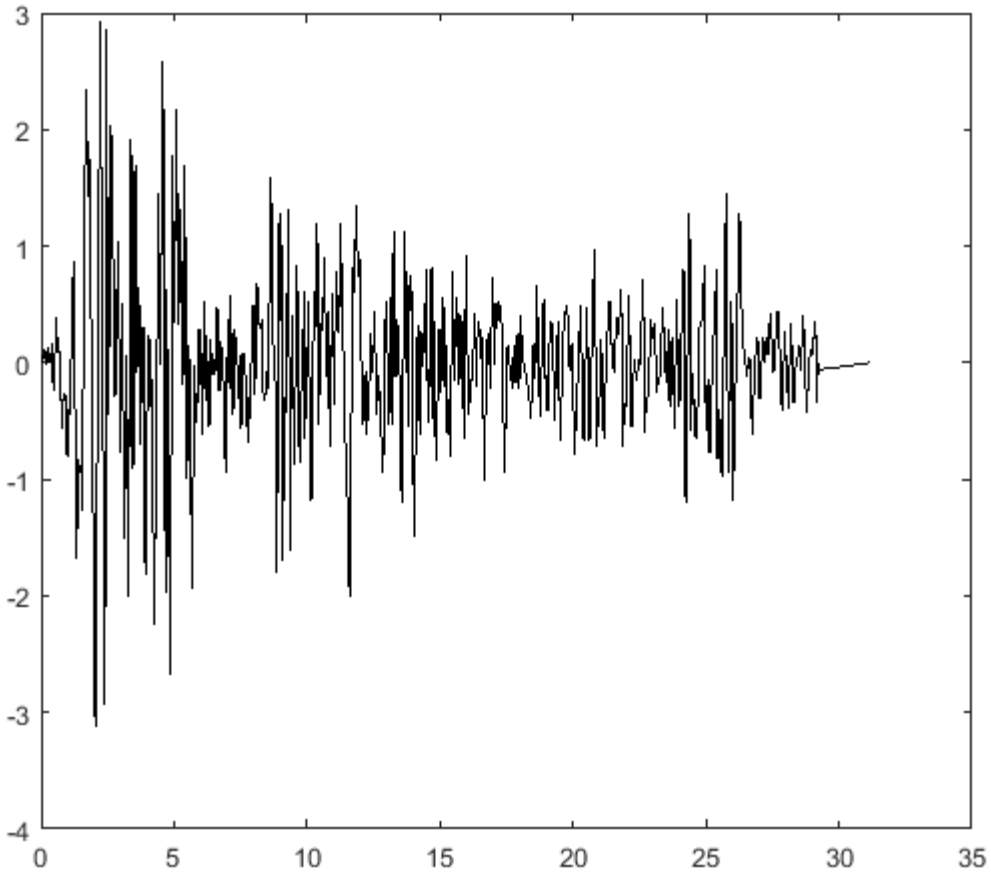

```
figure(6)
plot(S2.time,S2.acc,'k','LineWidth',1)
```



Resample acceleration time history from 0.02 sec to 0.01 sec.

```
sw='resample';
dti=0.01;
S3=OpenSeismoMatlab(dt,xgtt,sw,dti);
```

```
figure(7)
plot(S3.time,S3.acc,'k','LineWidth',1)
```



PGA

```
sw='pga';
S4=OpenSeismoMatlab(dt,xgtt,sw);
```

```
S4.PGA
```

```
ans =

   3.127624200000000
```

PGV

```
sw='pgv';
S5=OpenSeismoMatlab(dt,xgtt,sw);
```

```
S5.PGV
```

```
ans =
```

```
    0.360920691000000
```

PGD

```
sw='pgd';
S6=OpenSeismoMatlab(dt,xgtt,sw);
```

```
S6.PGD
```

```
ans =

    0.211893410160001
```

Total cumulative energy, Arias intensity and significant duration

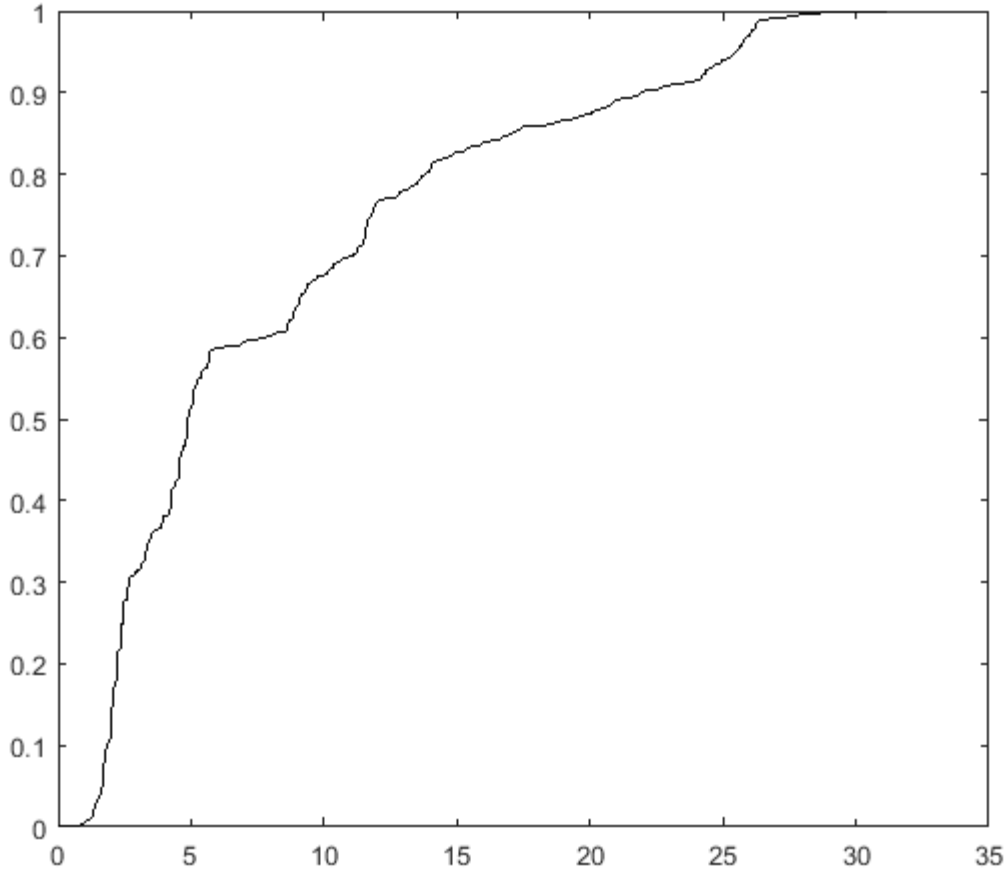
```
sw='arias';
S7=OpenSeismoMatlab(dt,xgtt,sw);
```

```
S7.Ecum
```

```
ans =

   11.251388628141965
```

```
figure(8)
plot(S7.time,S7.EcumTH,'k','LineWidth',1)
```




```
S7.t_5_95
```

```
ans =

   1.680000000000000  25.500000000000000
```

```
S7.Td_5_95
```

```
ans =

  23.840000000000000
```

```
S7.t_5_75
```

```
ans =

   1.680000000000000  11.779999999999999
```

```
S7.Td_5_75
```

```
ans =

  10.119999999999999
```

```
S7.arias
```

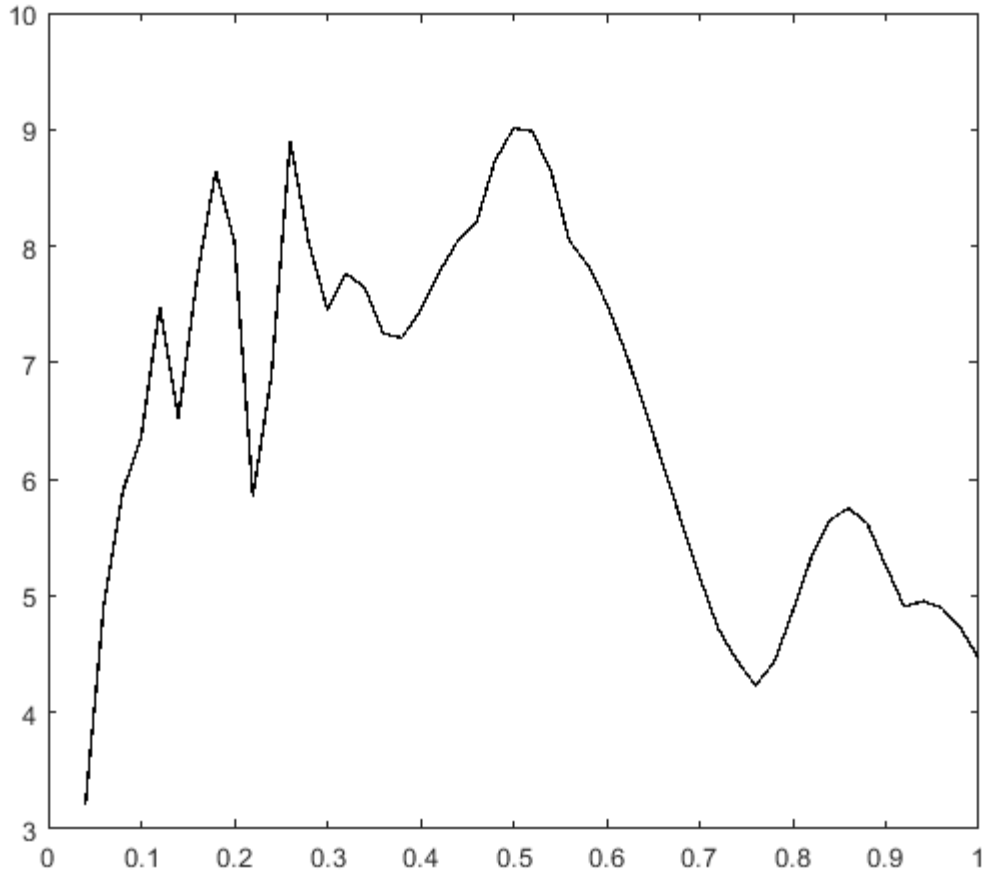
```
ans =

   1.645606908074047
```

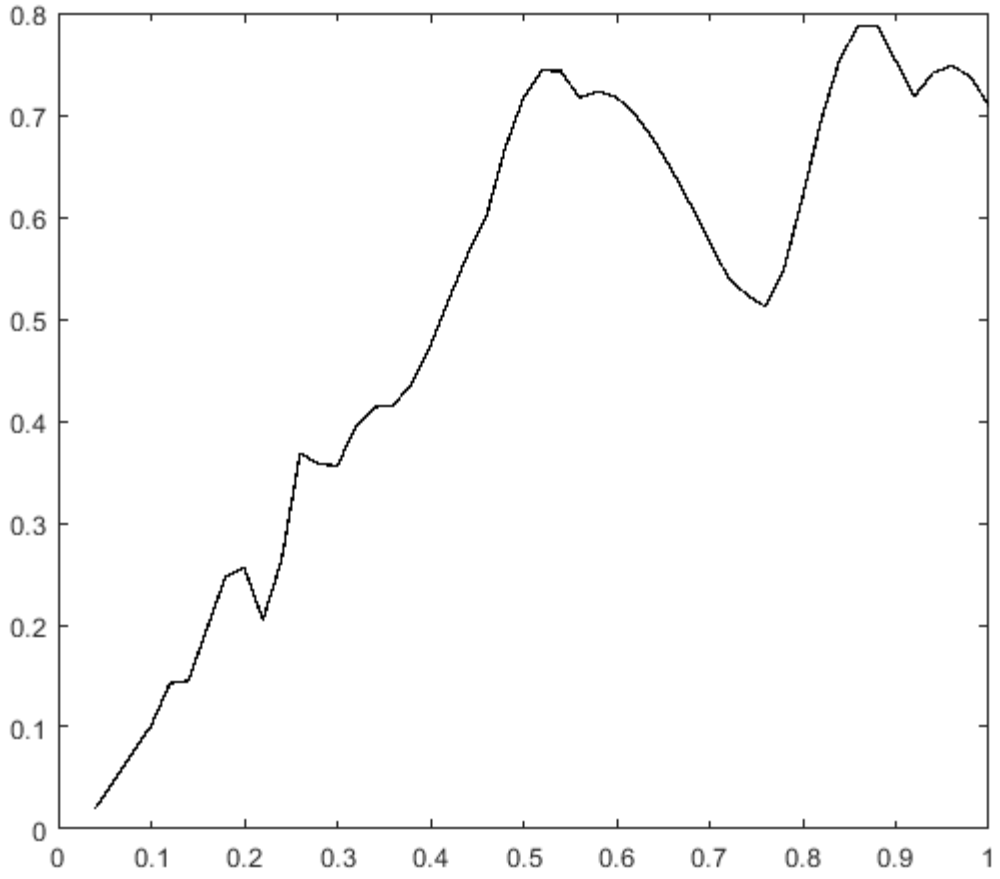
Linear elastic response spectra and pseudospectra

```
sw='es';
ksi=0.05;
T=0.04:0.02:1;
S8=OpenSeismoMatlab(dt,xgtt,sw,T,ksi);
```

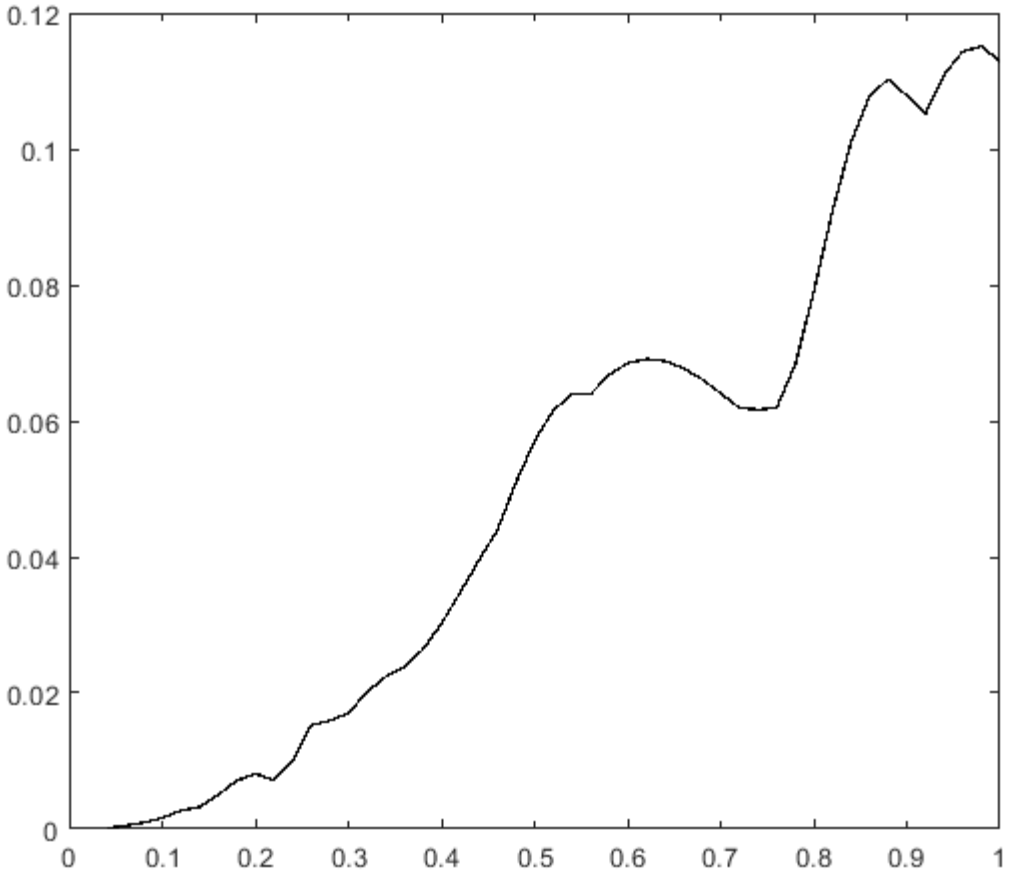
```
figure(9)
plot(S8.Period,S8.PSa,'k','LineWidth',1)
```



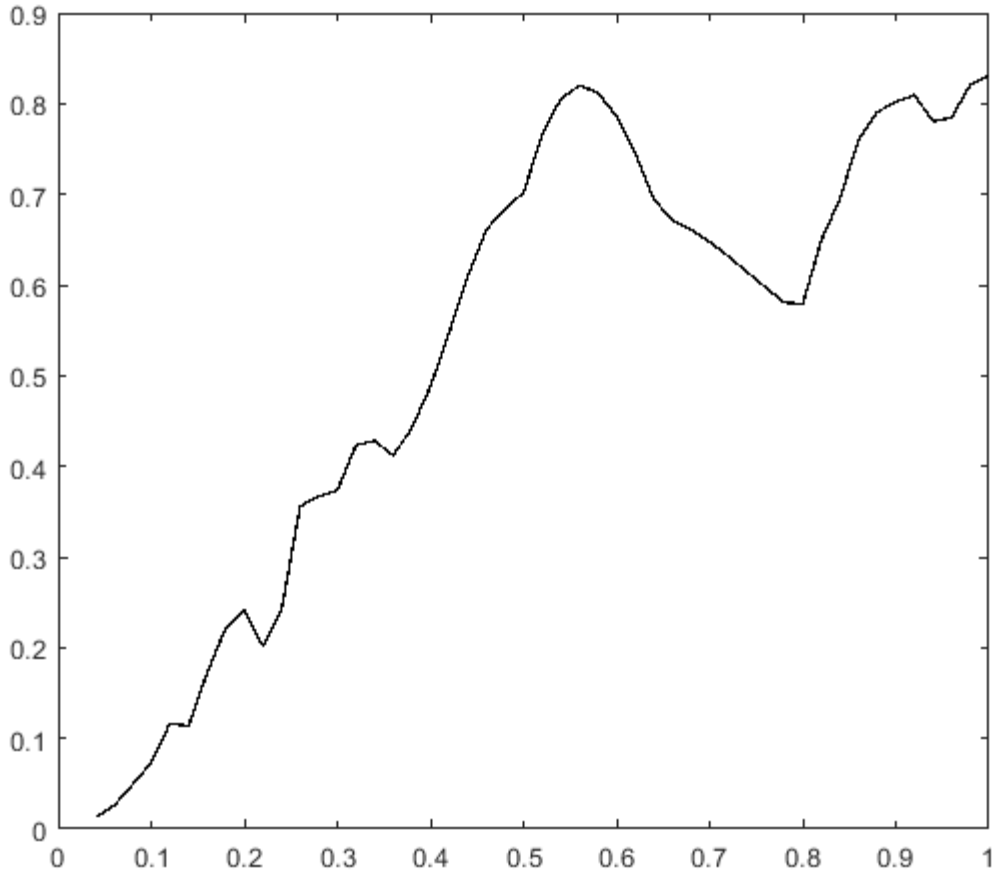

```
figure(10)
plot(S8.Period,S8.PSv,'k','LineWidth',1)
```



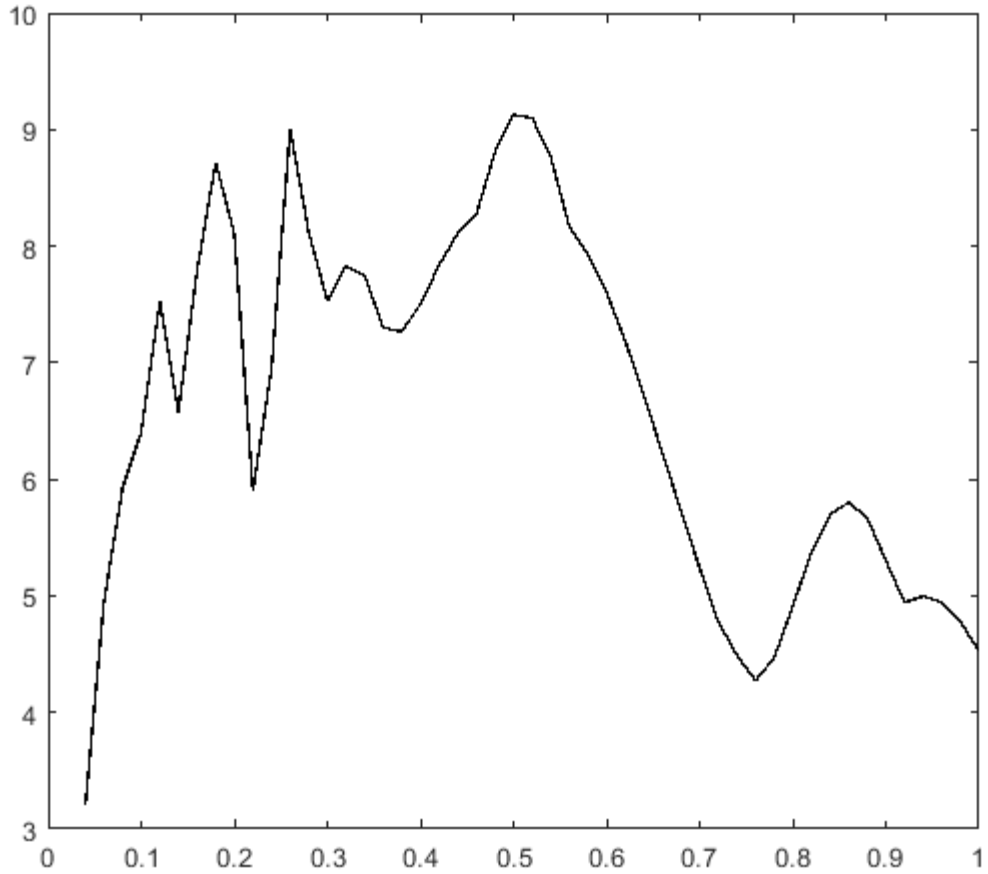

```
figure(11)
plot(S8.Period,S8.Sd,'k','LineWidth',1)
```



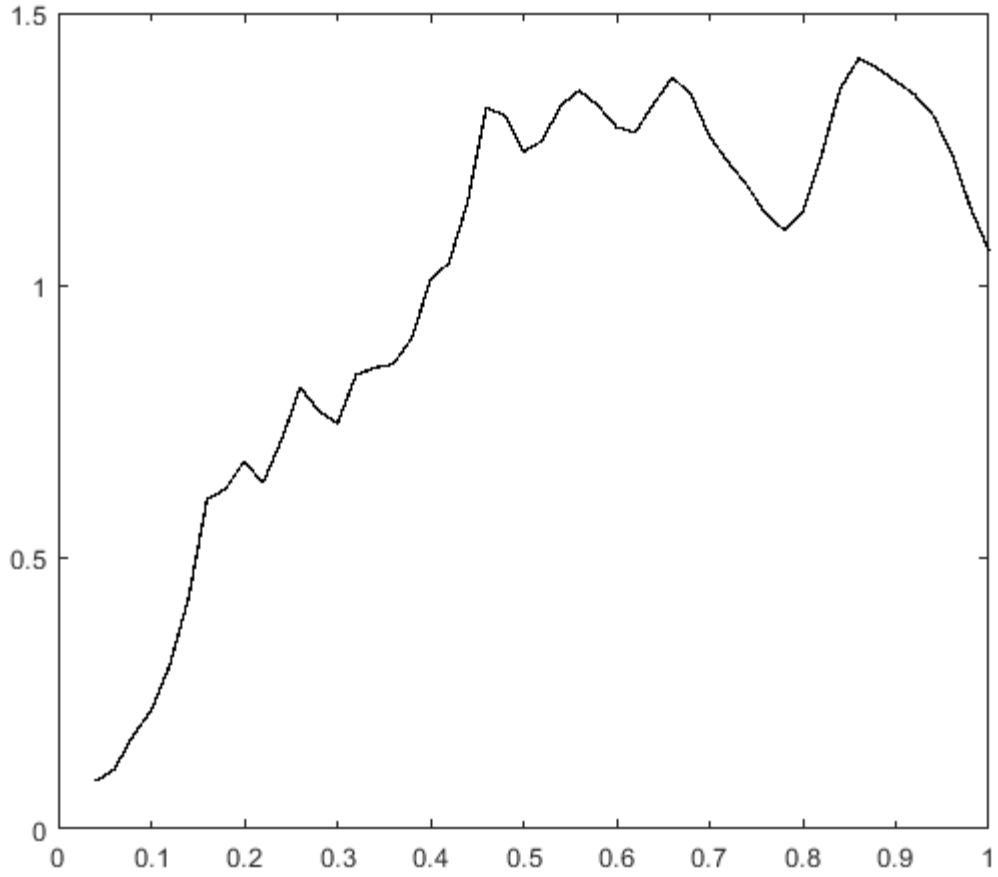

```
figure(12)
plot(S8.Period,S8.Sv,'k','LineWidth',1)
```




```
figure(13)
plot(S8.Period,S8.Sa,'k','LineWidth',1)
```

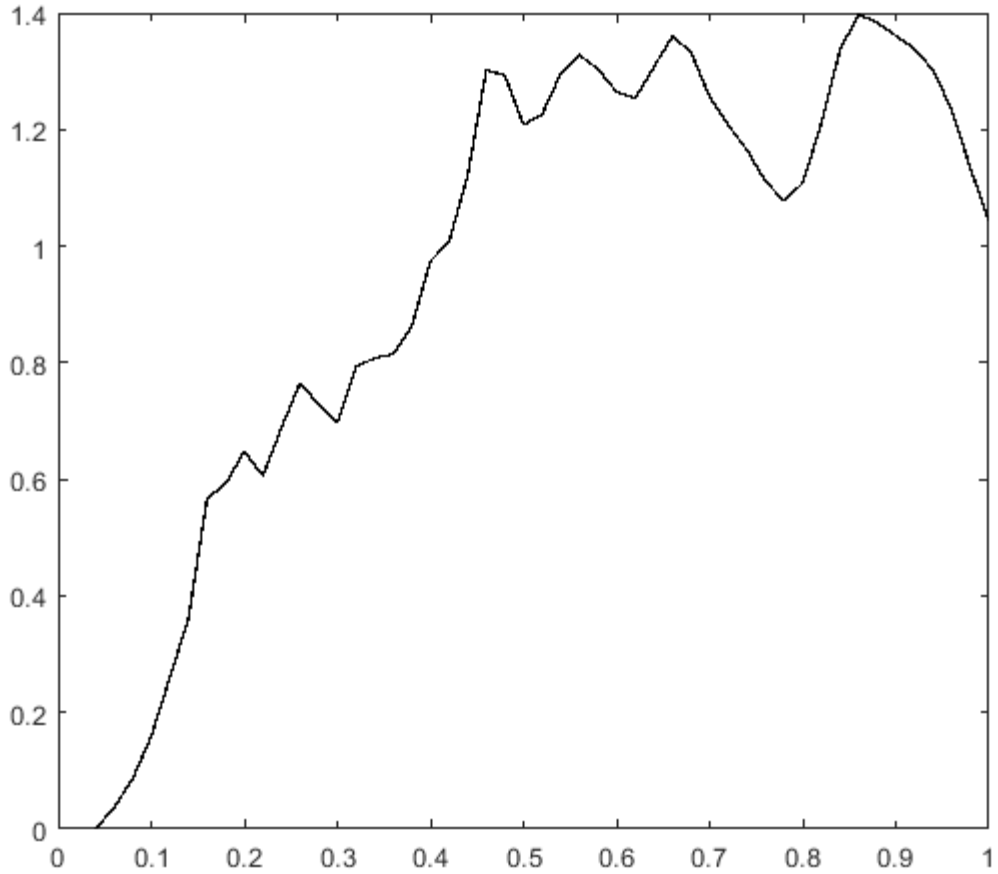



```
figure(14)
plot(S8.Period,S8.SievABS,'k','LineWidth',1)
```




```
figure(15)
plot(S8.Period,S8.SievREL,'k','LineWidth',1)
```

```
Warning: Imaginary parts of complex X
and/or Y arguments ignored
```




```
S8.PredPSa
```

```
ans =

   9.011815256159021
```

```
S8.PredPeriod
```

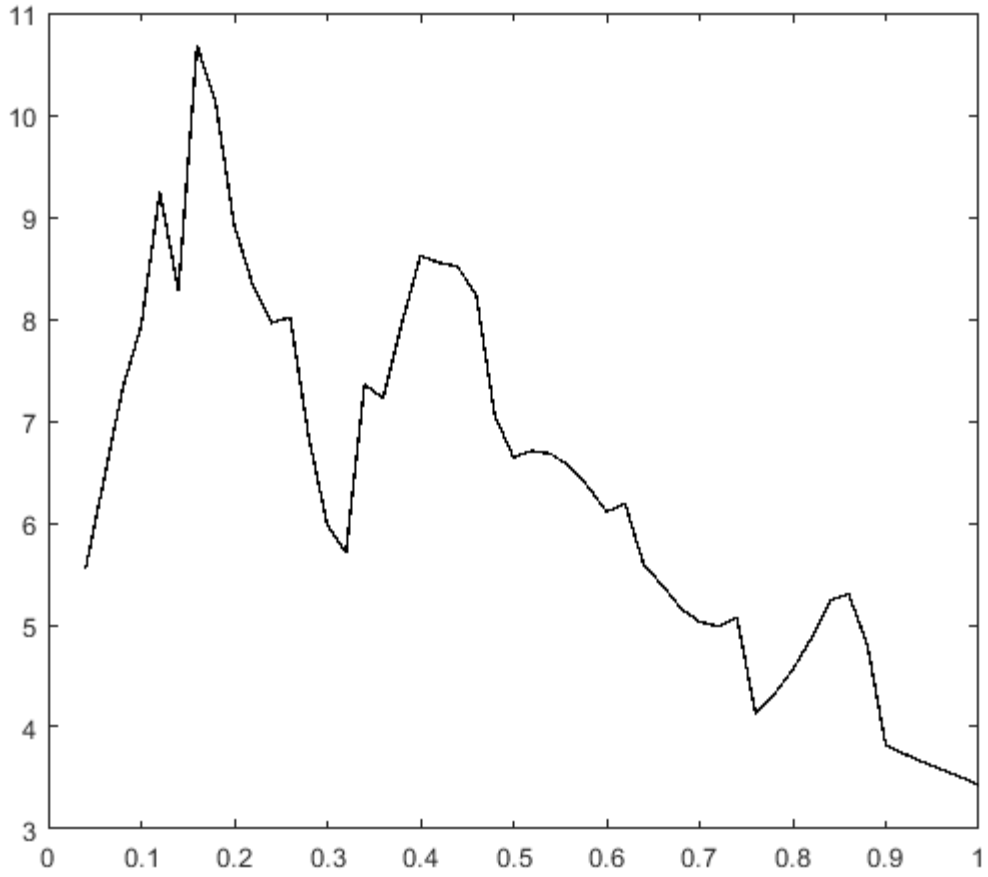
```
ans =

   0.500000000000000
```

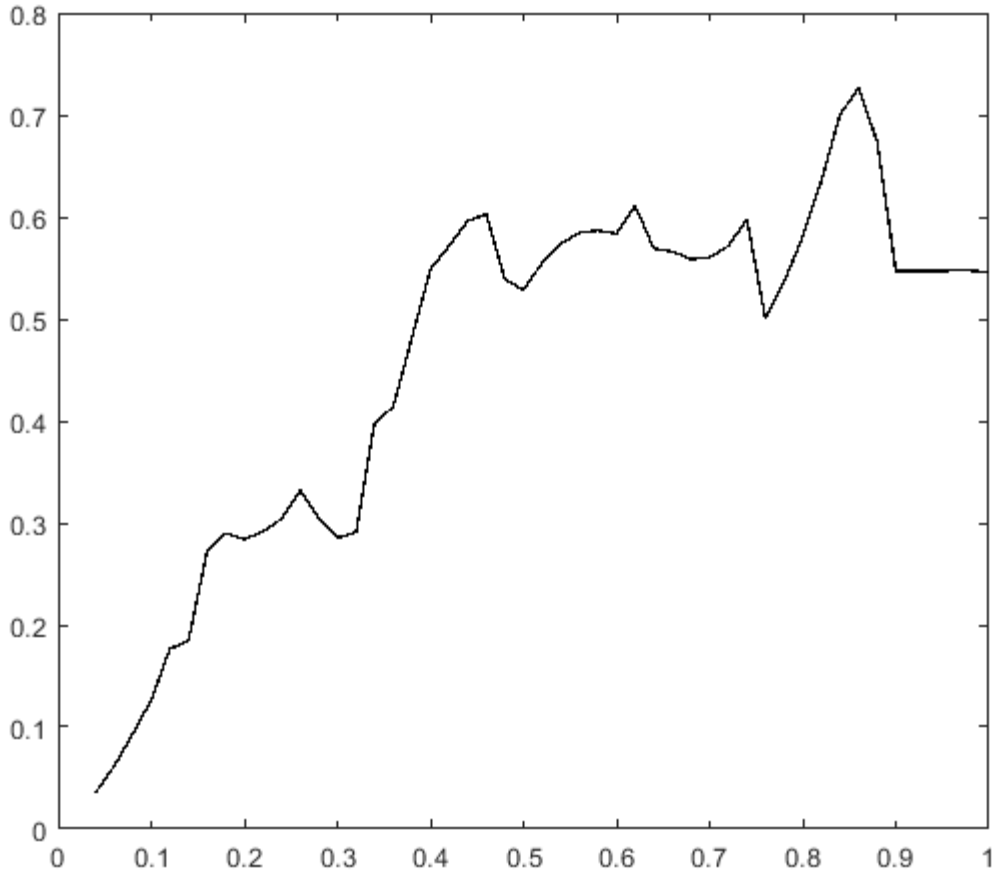
Constant ductility response spectra and pseudospectra

```
sw='cds';
ksi=0.05;
T=0.04:0.02:1;
mu=2;
S9=OpenSeismoMatlab(dt,xgtt,sw,T,ksi,mu);
```

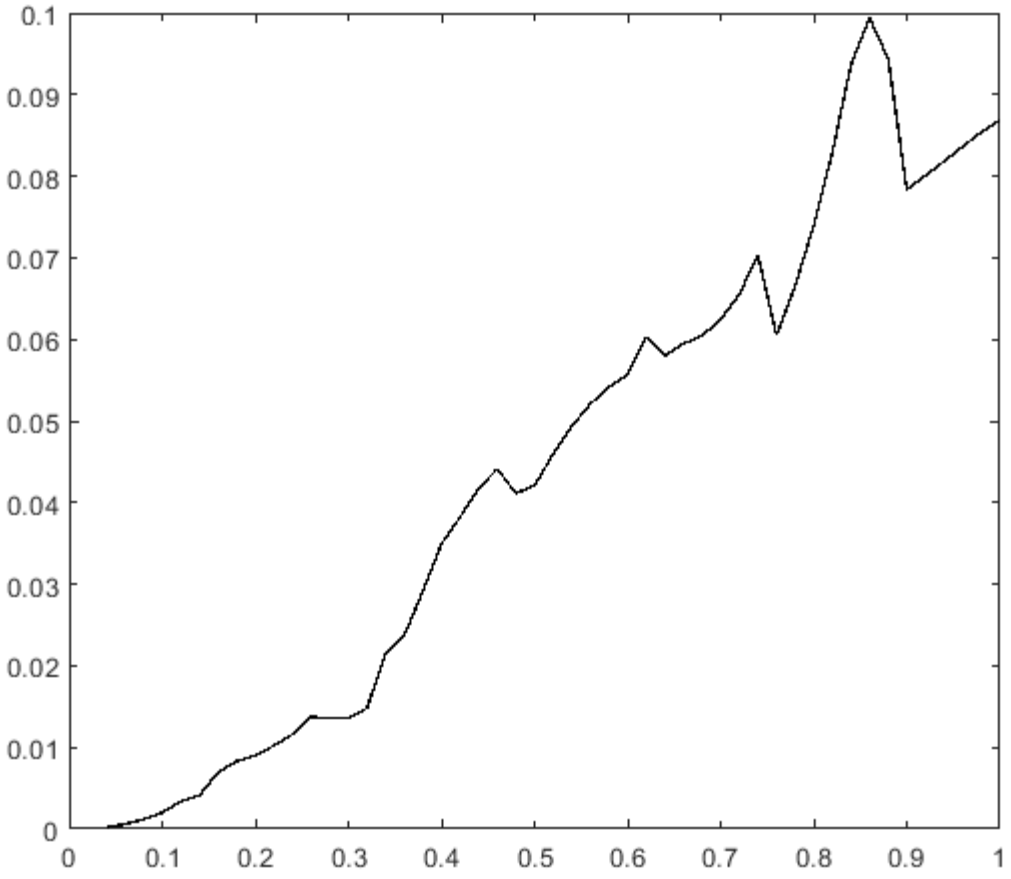
```
figure(15)
plot(S9.Period,S9.CDPSa,'k','LineWidth',1)
```



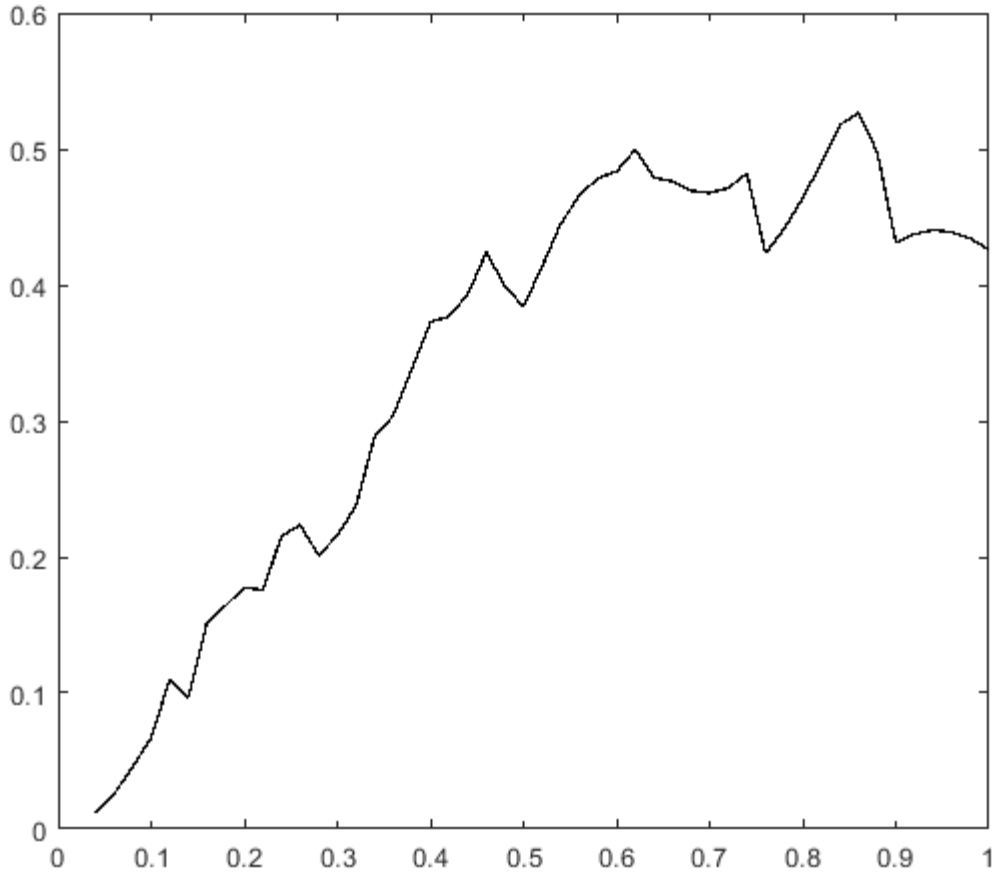

```
figure(16)
plot(S9.Period,S9.CDPSv,'k','LineWidth',1)
```



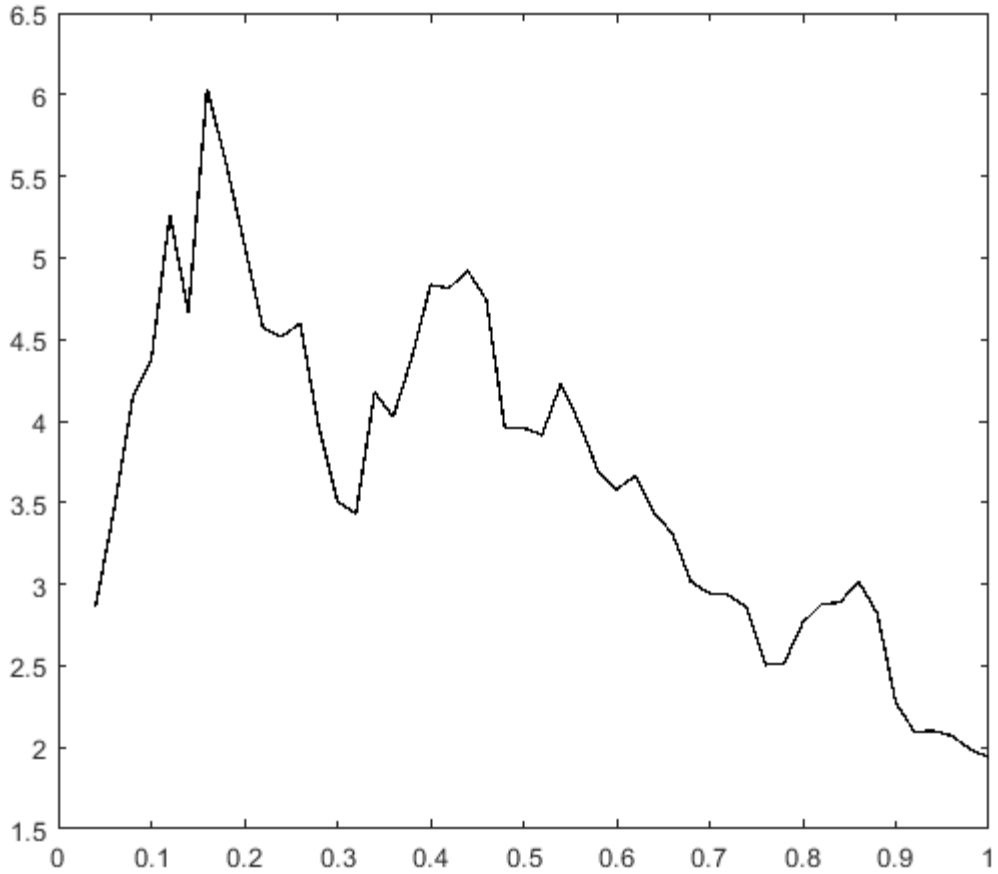

```
figure(17)
plot(S9.Period,S9.CDSd,'k','LineWidth',1)
```



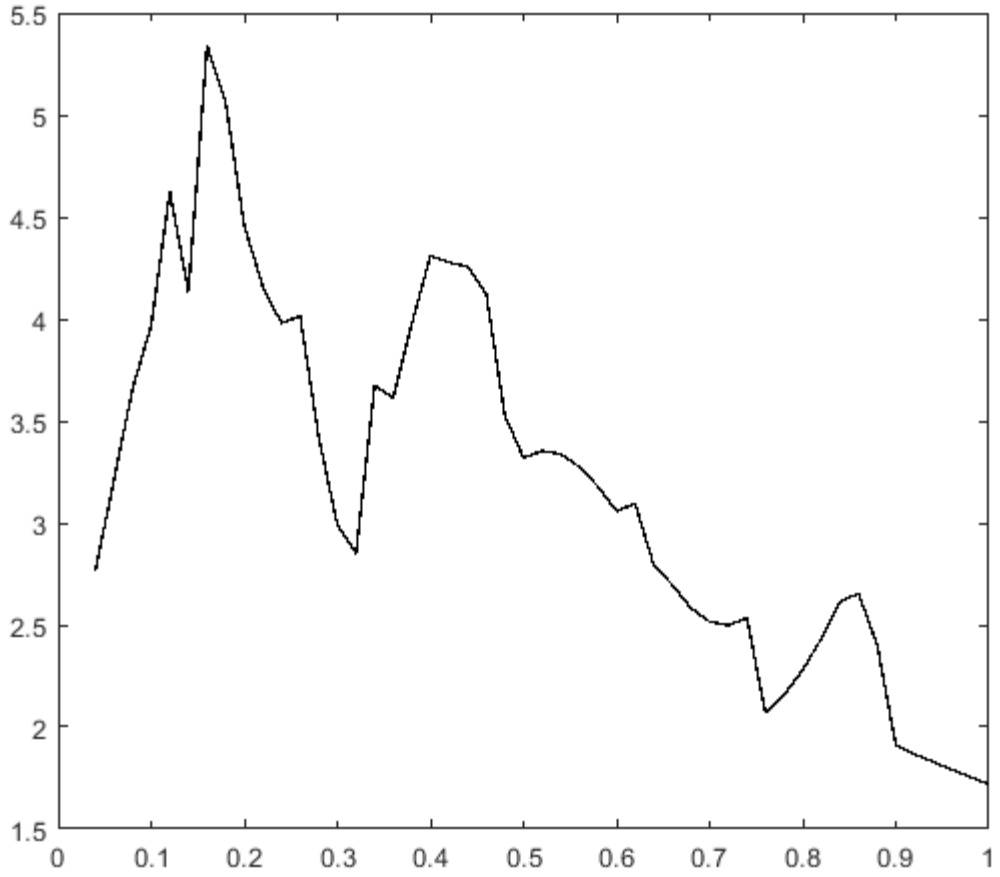

```
figure(18)
plot(S9.Period,S9.CDSv,'k','LineWidth',1)
```



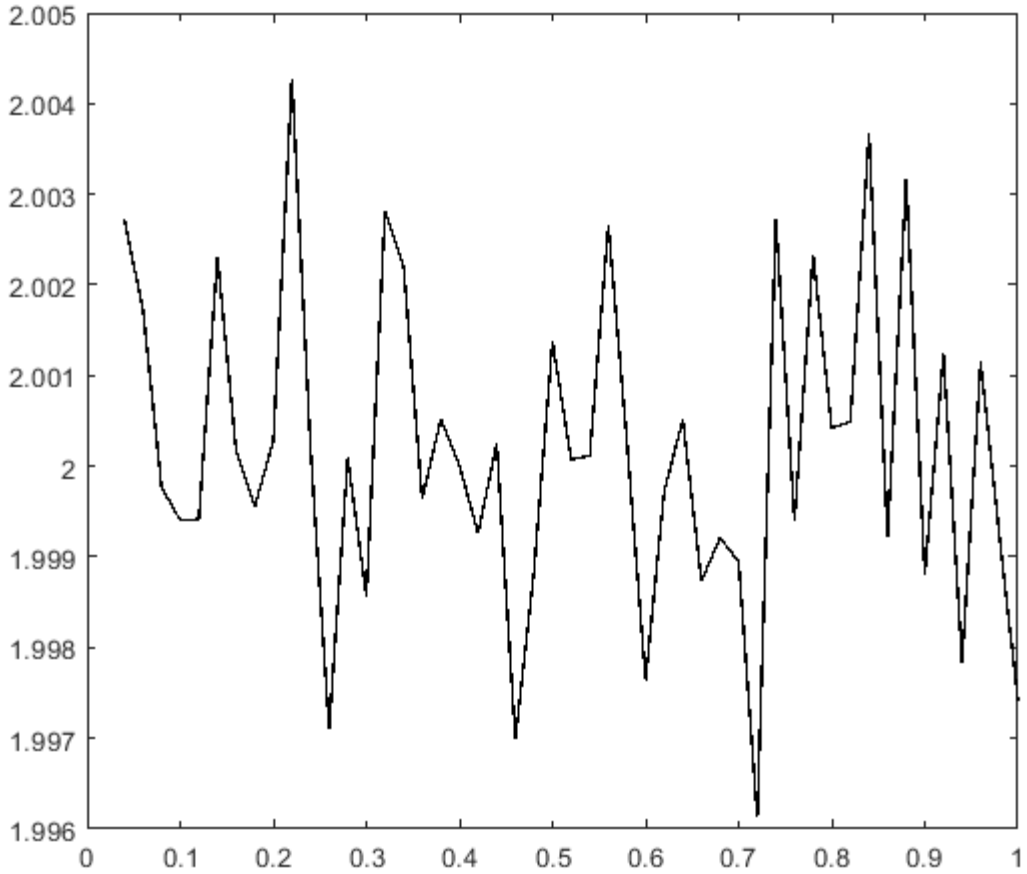

```
figure(19)
plot(S9.Period,S9.CDSa,'k','LineWidth',1)
```



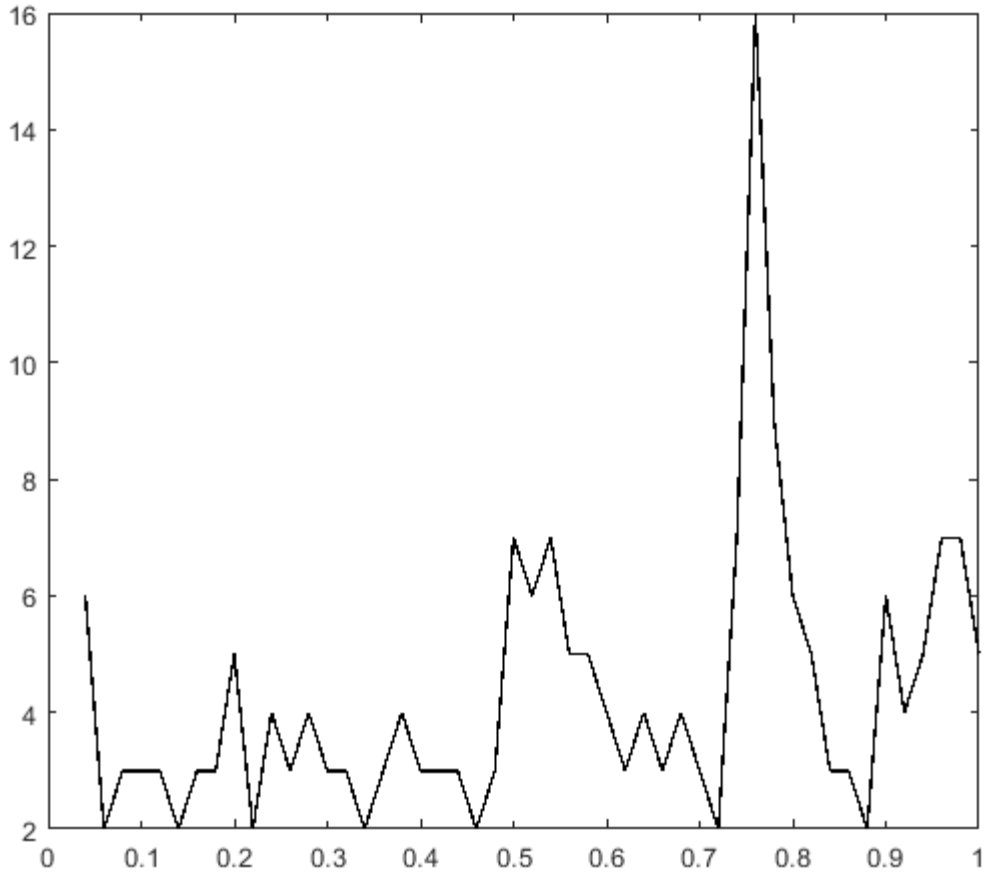

```
figure(20)
plot(S9.Period,S9.fyK,'k','LineWidth',1)
```




```
figure(21)
plot(S9.Period,S9.muK,'k','LineWidth',1)
```



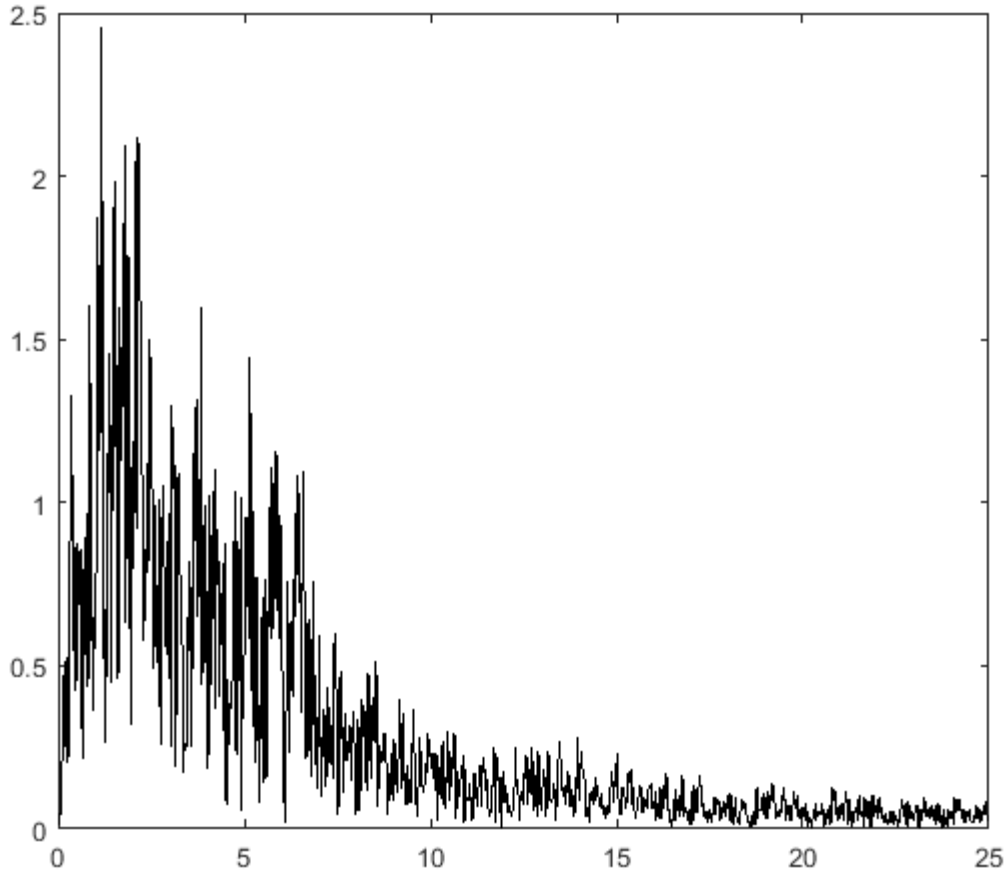

```
figure(22)
plot(S9.Period,S9.iterK,'k','LineWidth',1)
```



Fourier amplitude spectrum and mean period

```
sw='fs';
S10=OpenSeismoMatlab(dt,xgtt,sw);
```

```
figure(23)
plot(S10.freq,S10.FAS,'k','LineWidth',1)
```




```
S10.Tm
```

```
ans =

   0.533389590787339
```

```
S10.Fm
```

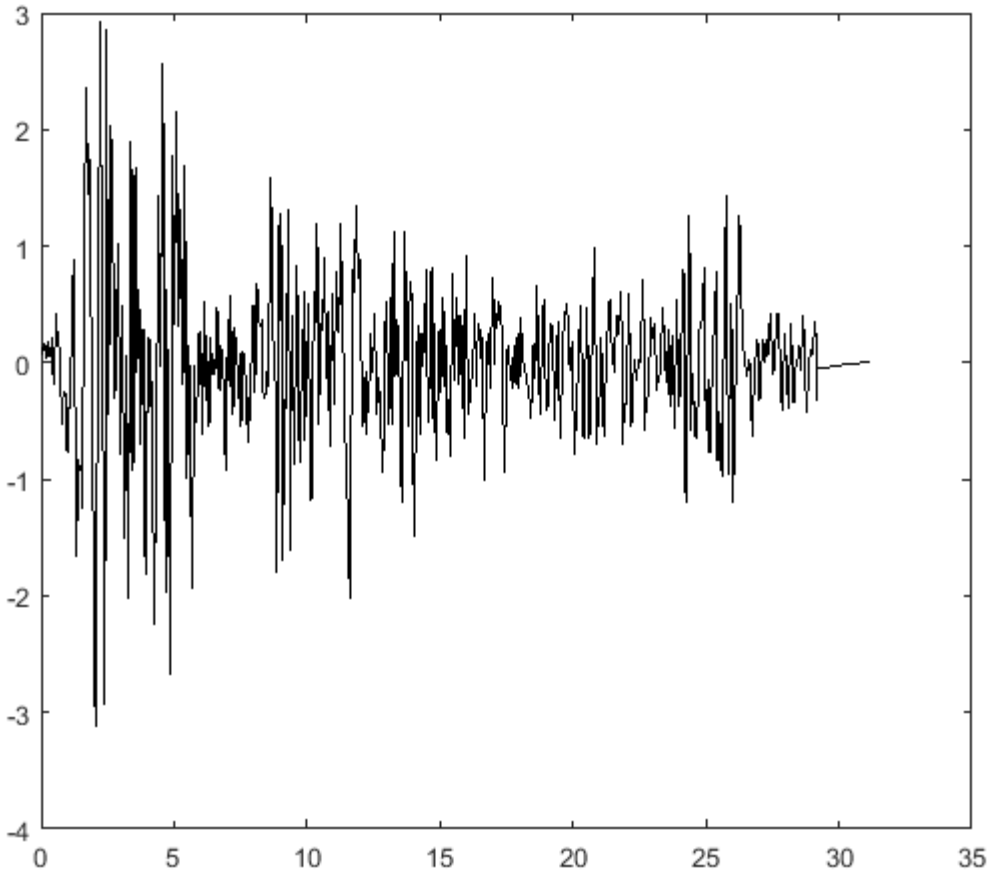
```
ans =

   3.293755163661825
```

High pass Butterworth filter

```
sw='butterworthhigh';
bOrder=4;
flc=0.1;
S11=OpenSeismoMatlab(dt,xgtt,sw,bOrder,flc);
```

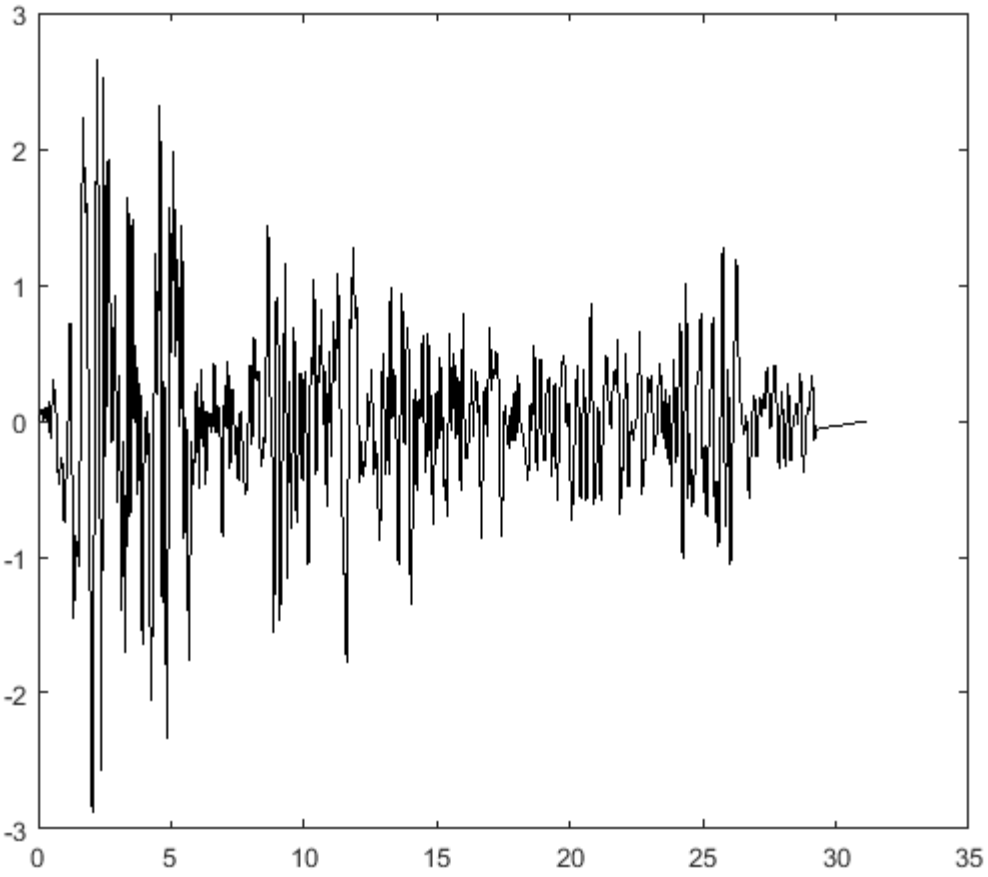
```
figure(24)
plot(S11.time,S11.acc,'k','LineWidth',1)
```



Low pass Butterworth filter

```
sw='butterworthlow';
bOrder=4;
fhc=10;
S12=OpenSeismoMatlab(dt,xgtt,sw,bOrder,fhc);
```

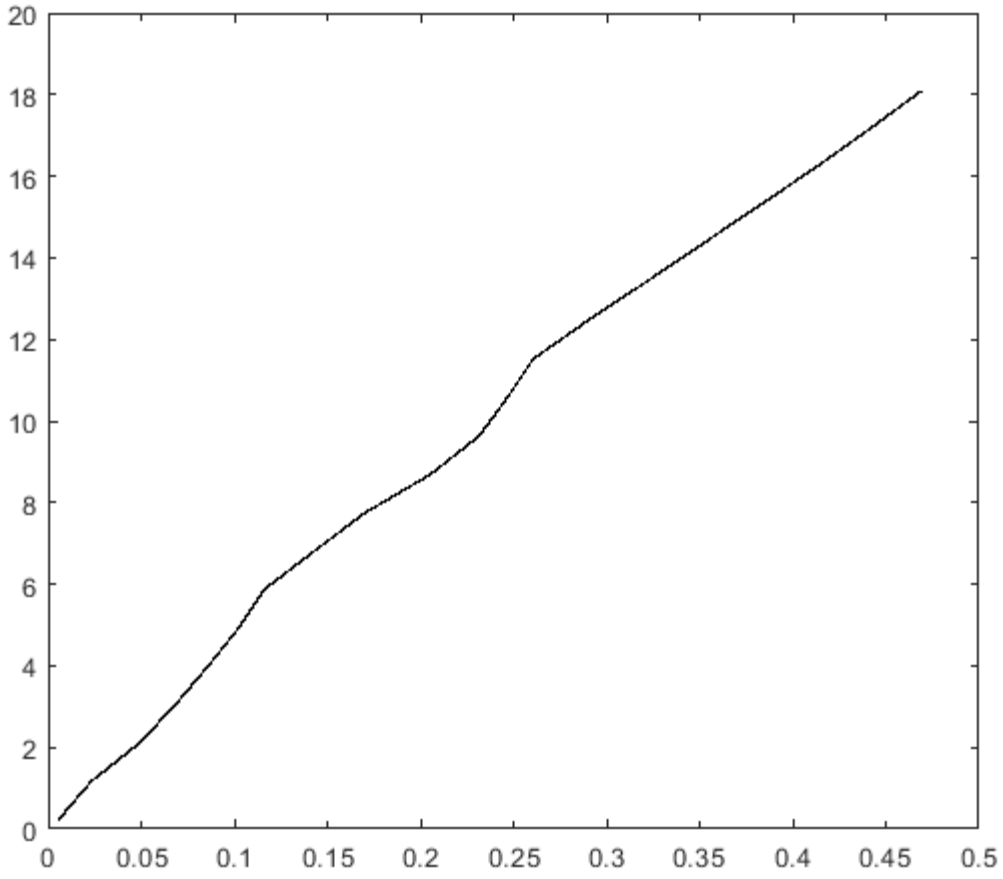
```
figure(25)
plot(S12.time,S12.acc,'k','LineWidth',1)
```



Incremental dynamic analysis

```
sw='ida';
S13=OpenSeismoMatlab(dt,xgtt,sw);
```

```
figure(26)
plot(S13.DM,S13.IM,'k','LineWidth',1)
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

LEReSp

Calculate linear elastic response spectra in OpenSeismoMatlab

Contents

Generate earthquake motion

For reproducibility

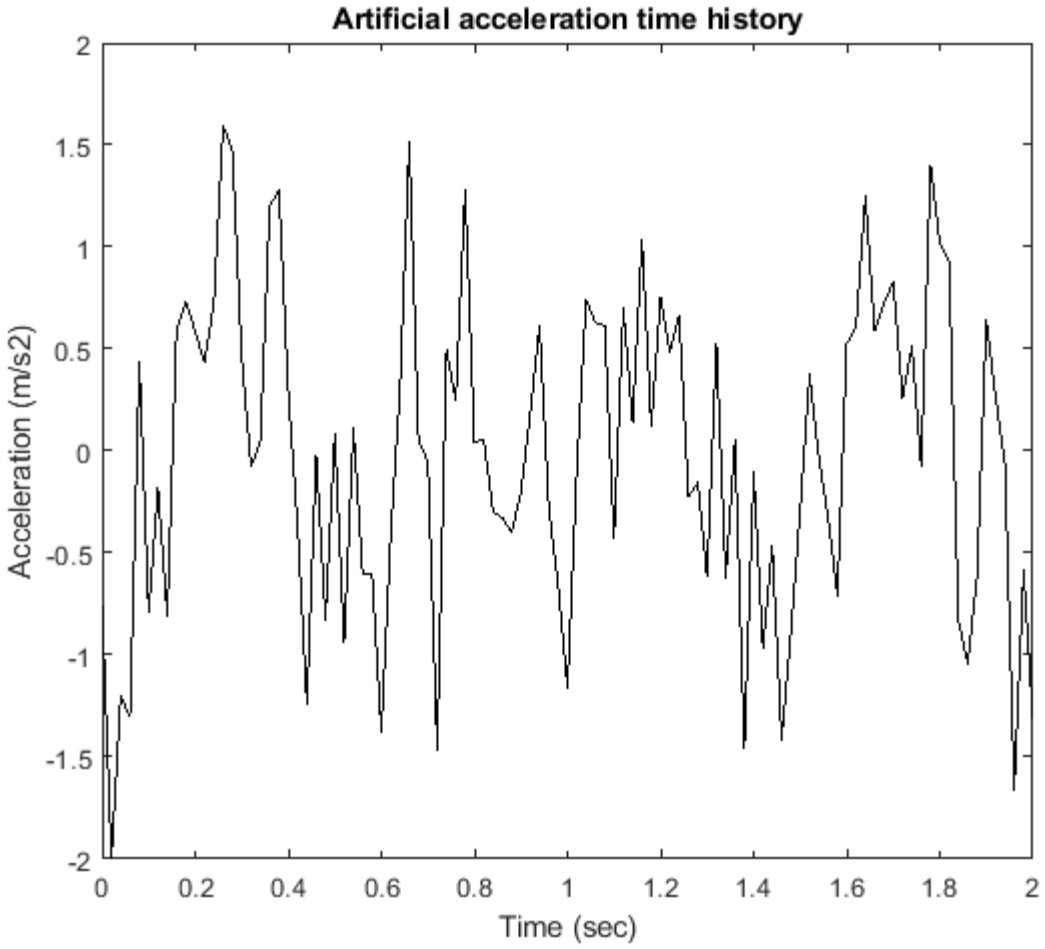
```
rng(0)
```

Generate earthquake acceleration time history

```
dt=0.02;
N=10;
a=rand(N,1)-0.5;
b=100*pi*rand(N,1);
c=pi*(rand(N,1)-0.5);
t=(0:dt:(100*dt))';
xgtt=zeros(size(t));
for i=1:N
    xgtt=xgtt+a(i)*sin(b(i)*t+c(i));
end
```

Plot the generated time history

```
figure()
plot(t,xgtt,'k','LineWidth',1)
ylabel('Acceleration (m/s2)')
xlabel('Time (sec)')
title('Artificial acceleration time history')
```



Setup parameters for LEReSp function

Eigenperiods

```
T=logspace(log10(0.02),log10(5),1000)';
```

Critical damping ratio

```
ksi=0.05;
```

Maximum ratio of the integration time step to the eigenperiod

```
dtTol=0.02;
```

Algorithm to be used for the time integration

```
AlgID='U0-V0-Opt';
```

Minimum absolute value of the eigenvalues of the amplification matrix

```
rinf=1;
```

Calculate spectra and pseudospectra

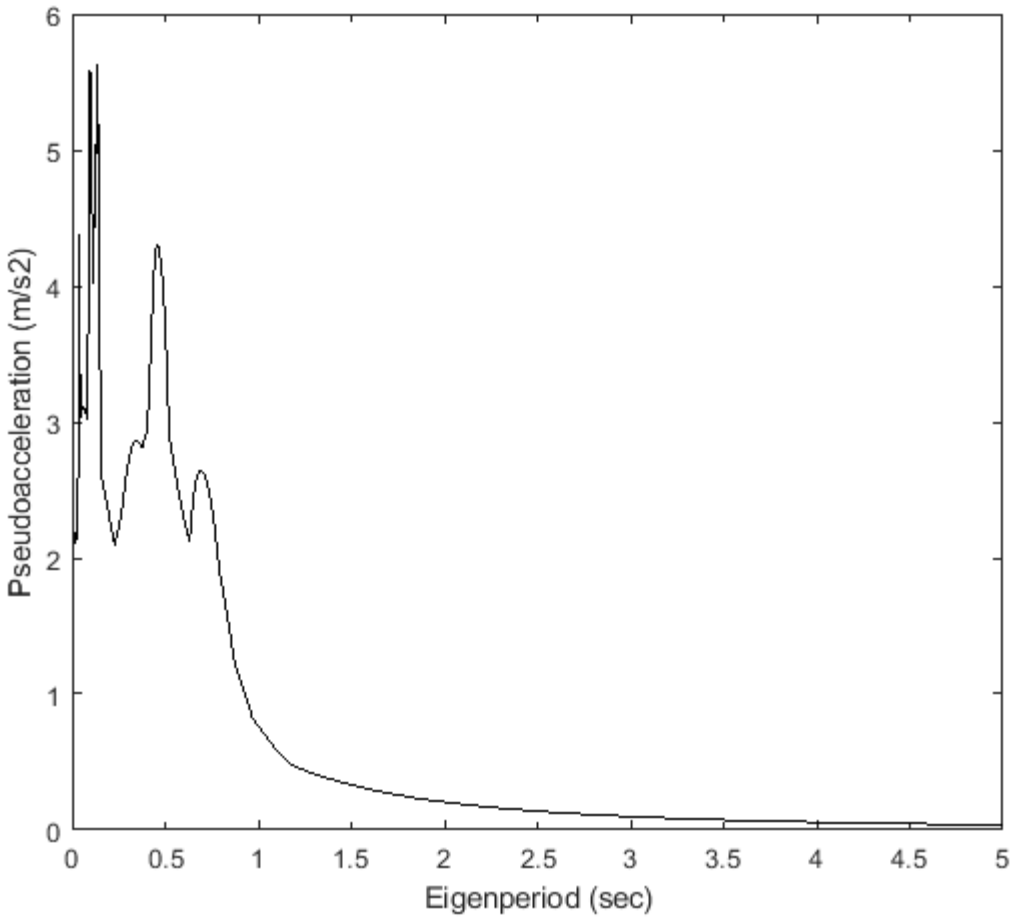
Apply LEReSp

```
[PSa,PSv,Sd,Sv,Sa,SievABS,SievREL]=LEReSp(dt,xgtt,T,ksi,dtTol,AlgID,rinf);
```

Plot the spectra and pseudospectra

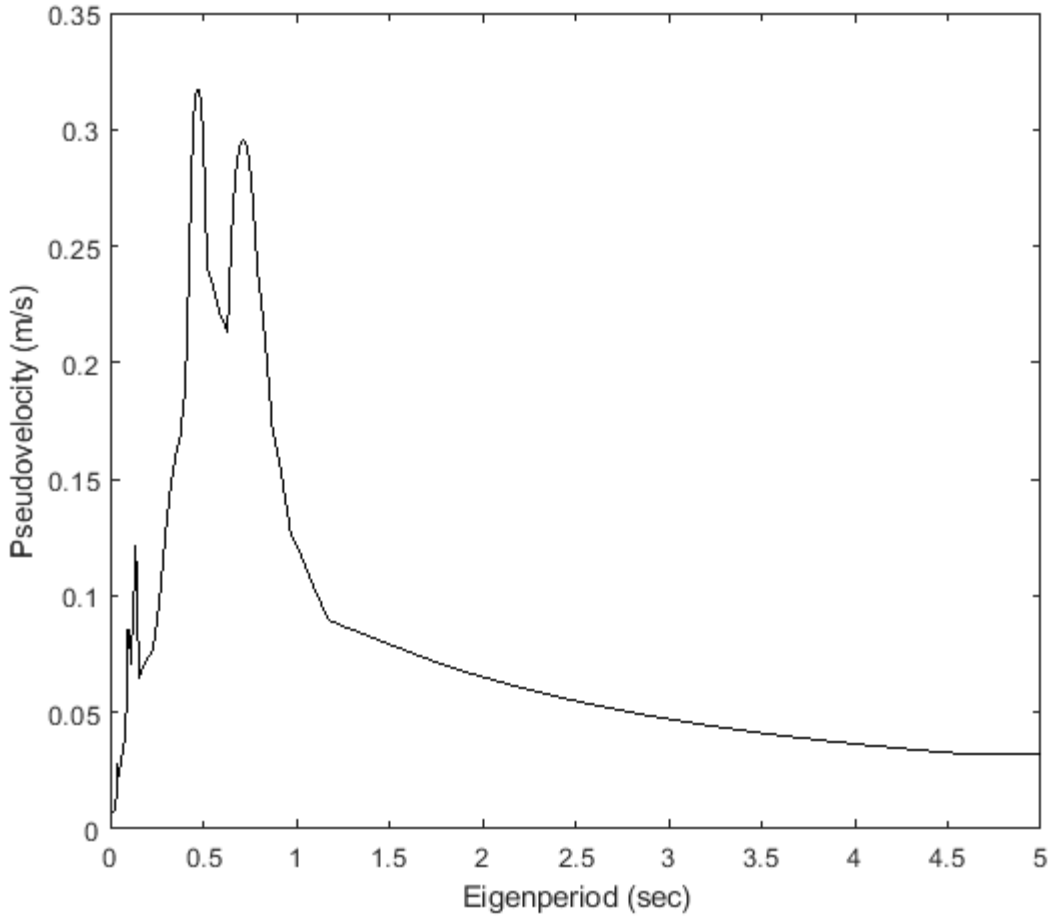
Pseudoacceleration spectrum

```
figure()
plot(T,PSa,'k','LineWidth',1)
ylabel('Pseudoacceleration (m/s2)')
xlabel('Eigenperiod (sec)')
```



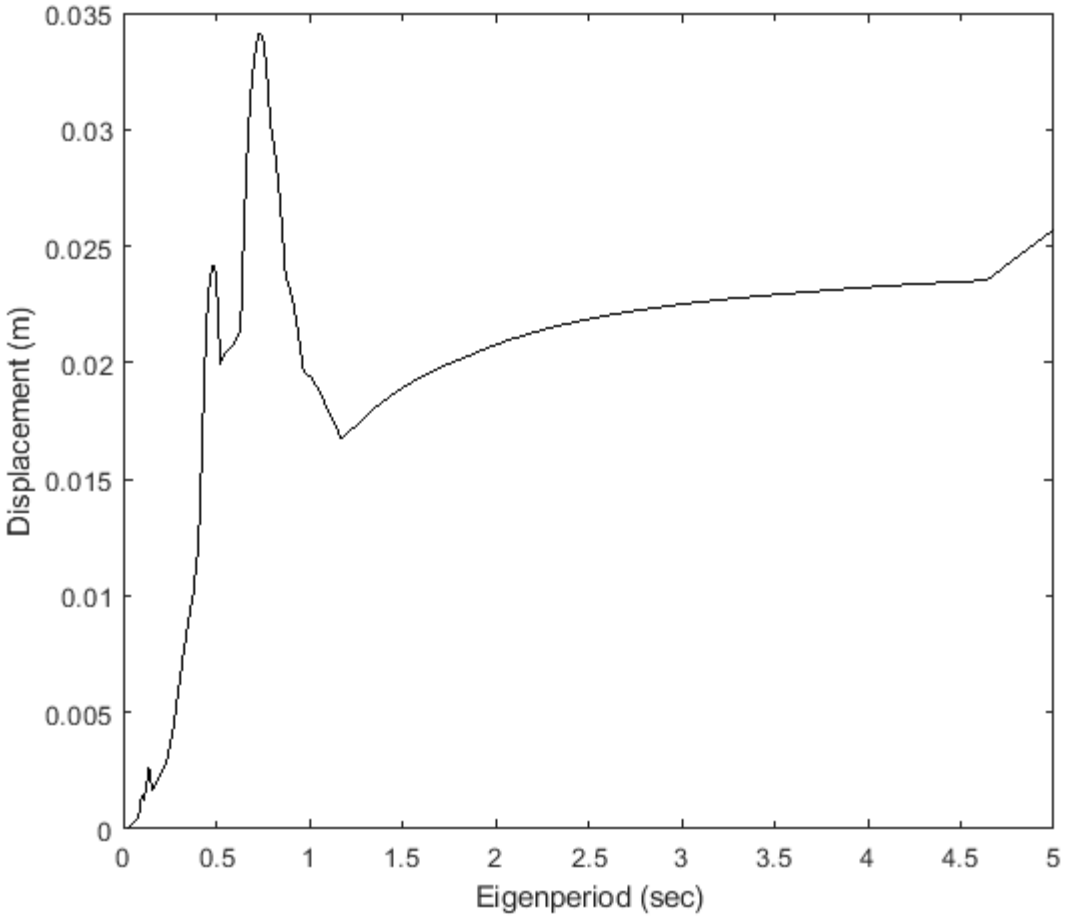
Pseudovelocity spectrum

```
figure()
plot(T,PSv,'k','LineWidth',1)
ylabel('Pseudovelocity (m/s)')
xlabel('Eigenperiod (sec)')
```



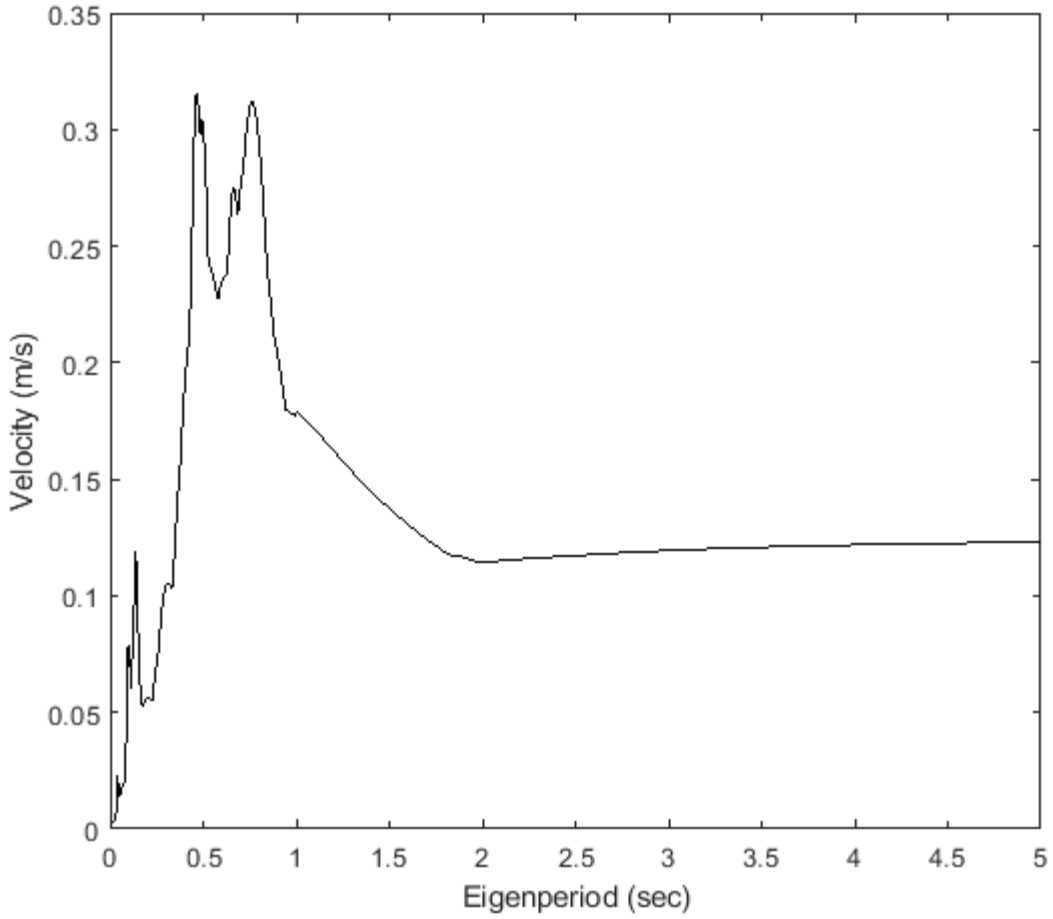
Displacement spectrum

```
figure()
plot(T,Sd,'k','LineWidth',1)
ylabel('Displacement (m)')
xlabel('Eigenperiod (sec)')
```



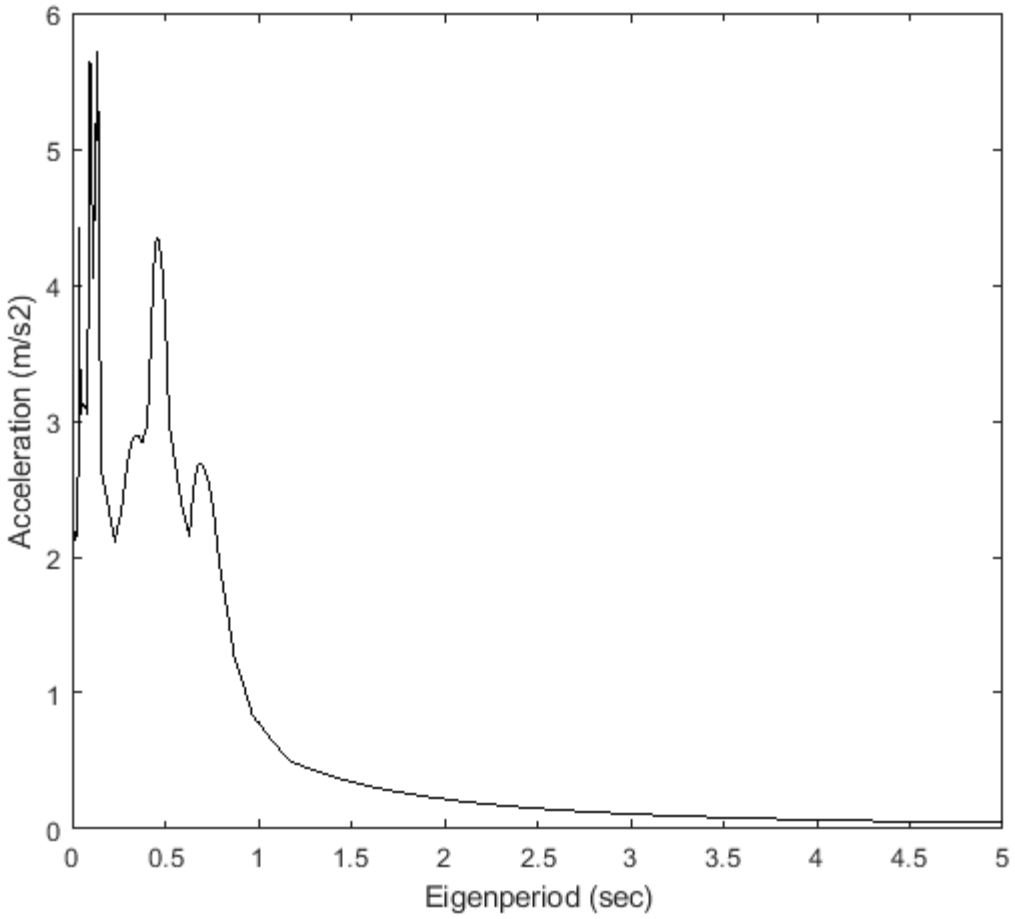
Velocity spectrum

```
figure()
plot(T,Sv,'k','LineWidth',1)
ylabel('Velocity (m/s)')
xlabel('Eigenperiod (sec)')
```



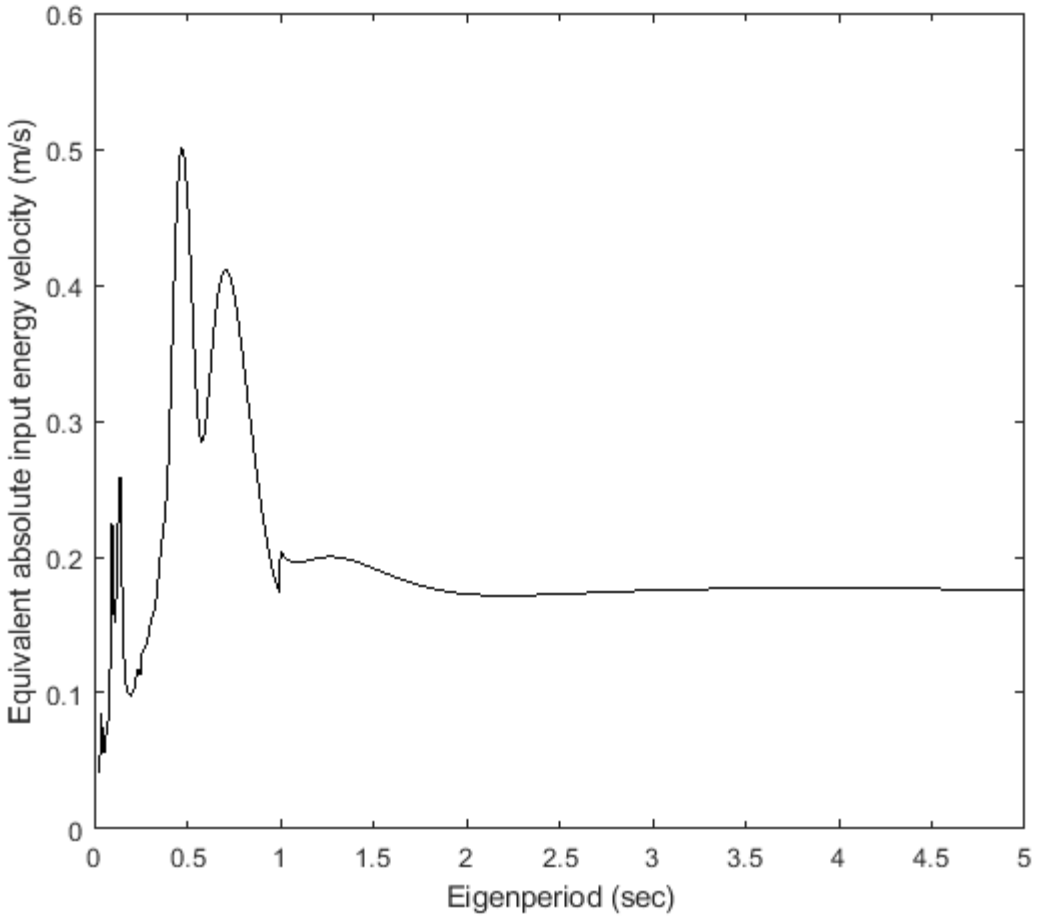
Acceleration spectrum

```
figure()
plot(T,Sa,'k','LineWidth',1)
ylabel('Acceleration (m/s2)')
xlabel('Eigenperiod (sec)')
```



Equivalent absolute input energy velocity

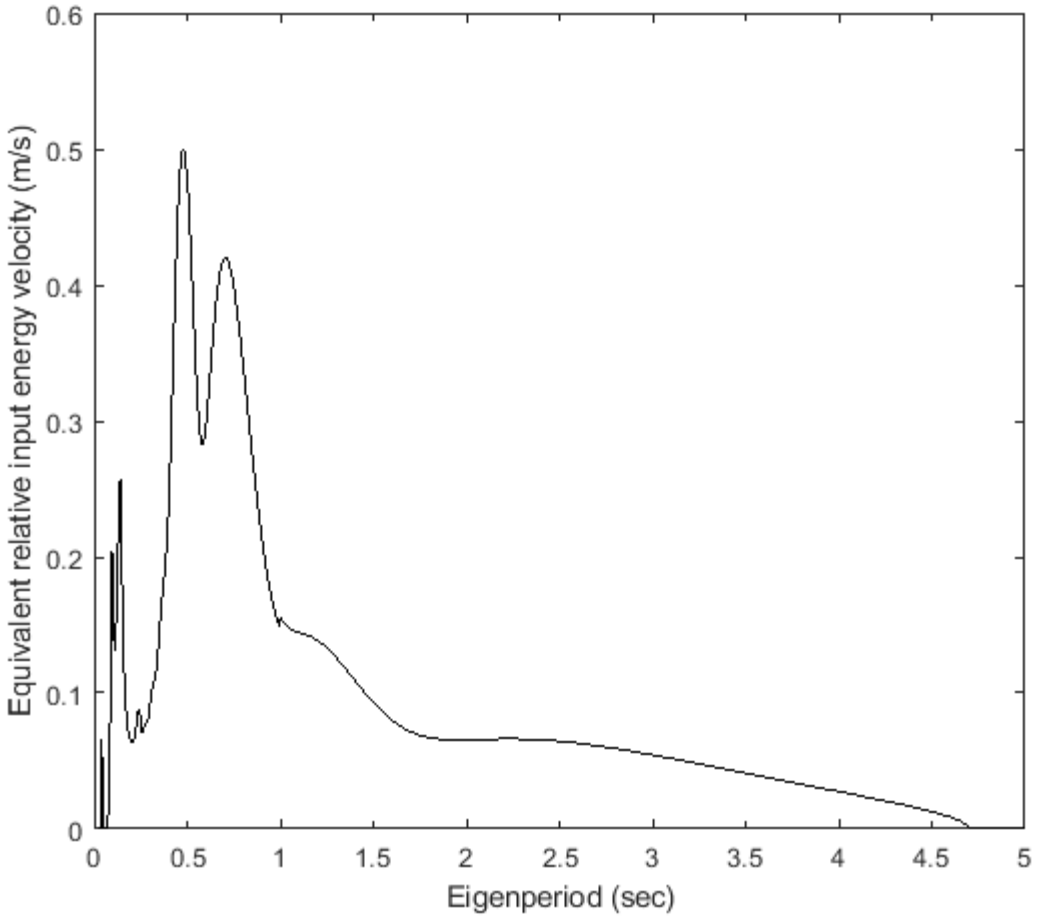
```
figure()
plot(T,SievABS,'k','LineWidth',1)
ylabel('Equivalent absolute input energy velocity (m/s)')
xlabel('Eigenperiod (sec)')
```



Equivalent relative input energy velocity

```
figure()
plot(T,SievREL,'k','LineWidth',1)
ylabel('Equivalent relative input energy velocity (m/s)')
xlabel('Eigenperiod (sec)')
```

```
Warning: Imaginary parts of complex X
and/or Y arguments ignored
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Linear elastic response spectra

Generate the linear elastic response spectra of an earthquake suite using OpenSeismoMatlab.

Contents

Input

Earthquake motions

```
eqmotions={'Imperial Valley'; % Imperial valley 1979
    'Kocaeli';
    'Loma Prieta';
    'Northridge';
    'San Fernando';
    'Spitak';
    'Cape Mendocino';
    'ChiChi';
    'El Centro'; % Imperial valley 1940
    'Hollister';
    'Kobe'};
```

Set the eigenperiod range for which the response spectra will be calculated.

```
Tspectra=(0.01:0.01:4)';
```

Set critical damping ratio of the response spectra to be calculated.

```
ksi=0.05;
```

Extract linear elastic response spectra

```
sw='es';
```

Calculation

Initialize LERS

```
LERS=cell(numel(eqmotions),1);
```

Calculation of peak values

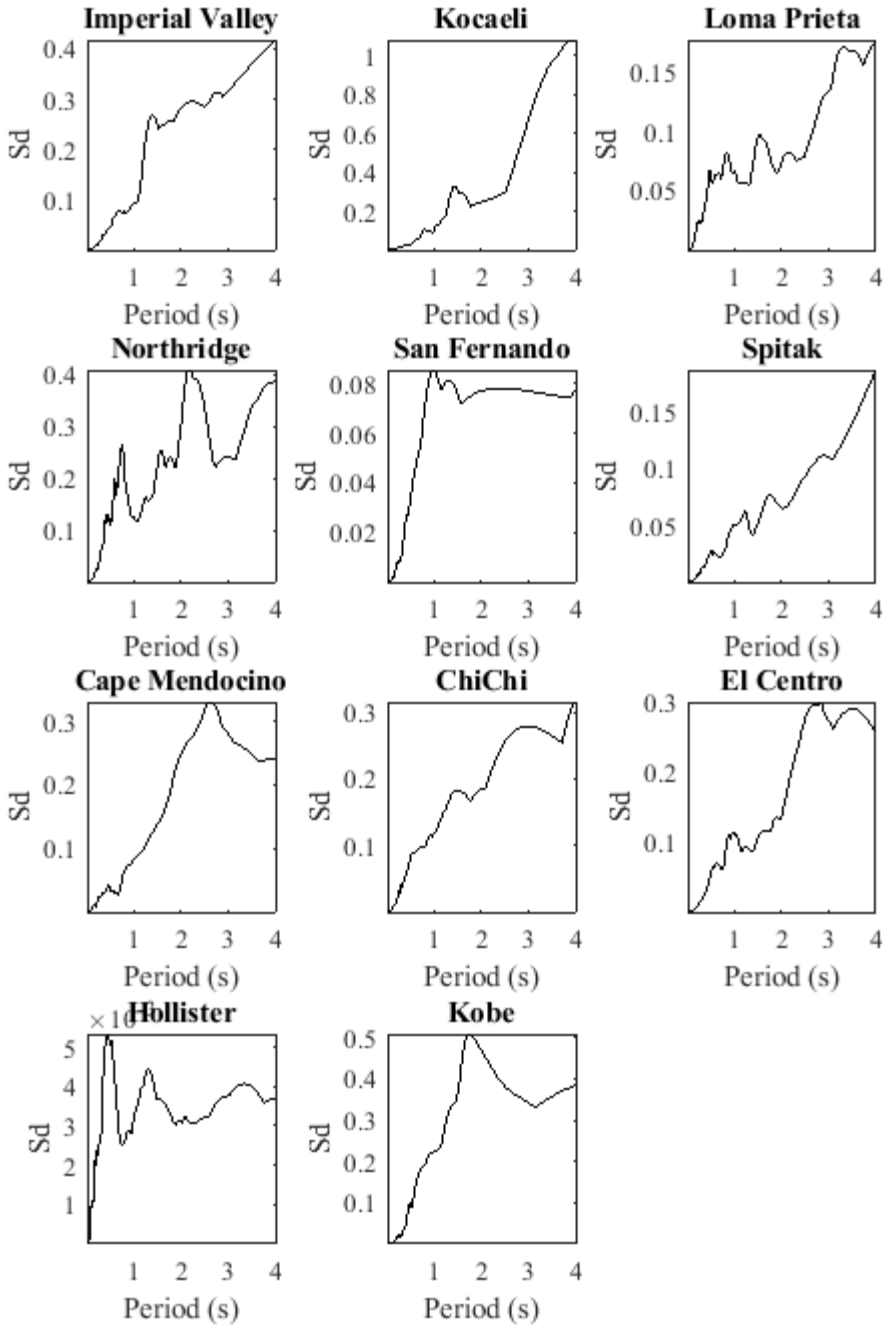
```
```

```
for i=1:numel(eqmotions)
    % earthquake
    data=load([eqmotions{i},'.dat']);
    t=data(:,1);
    dt=t(2)-t(1);
    xgtt=data(:,2);
    S=OpenSeismoMatlab(dt,xgtt,sw,Tspectra,ksi);
    LERS{i}=[S.Period,S.Sd,S.PSv,S.PSa];
end
```

Output

Plot displacement response spectra

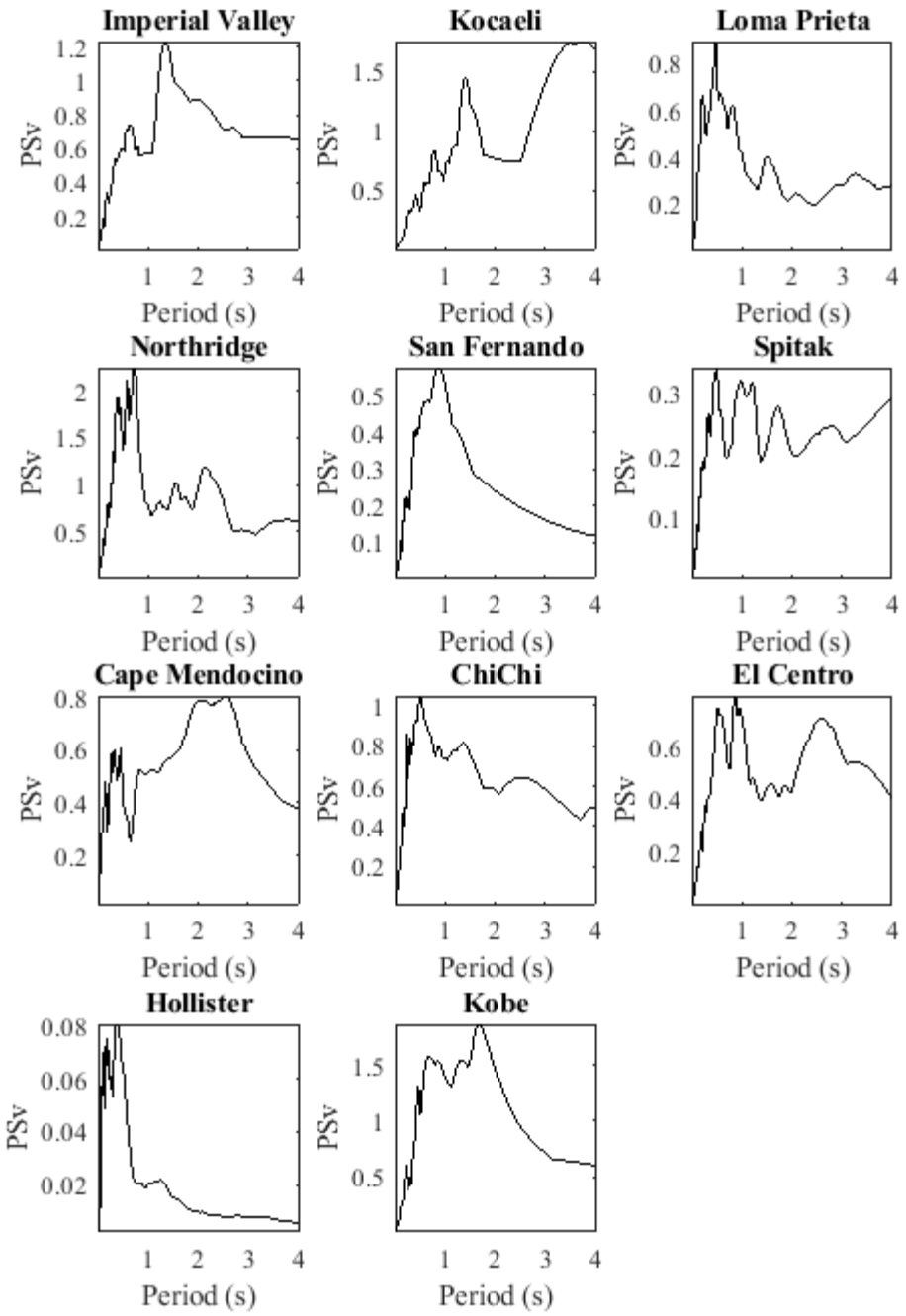
```
Fig1 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(LERS{i}(:,1),LERS{i}(:,2),'k','LineWidth',1);
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('Sd','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
    axis tight
end
```



Plot pseudo-velocity response spectra

```
Fig2 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(LERS{i}(:,1),LERS{i}(:,3),'k','LineWidth',1);
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('PSv','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
```

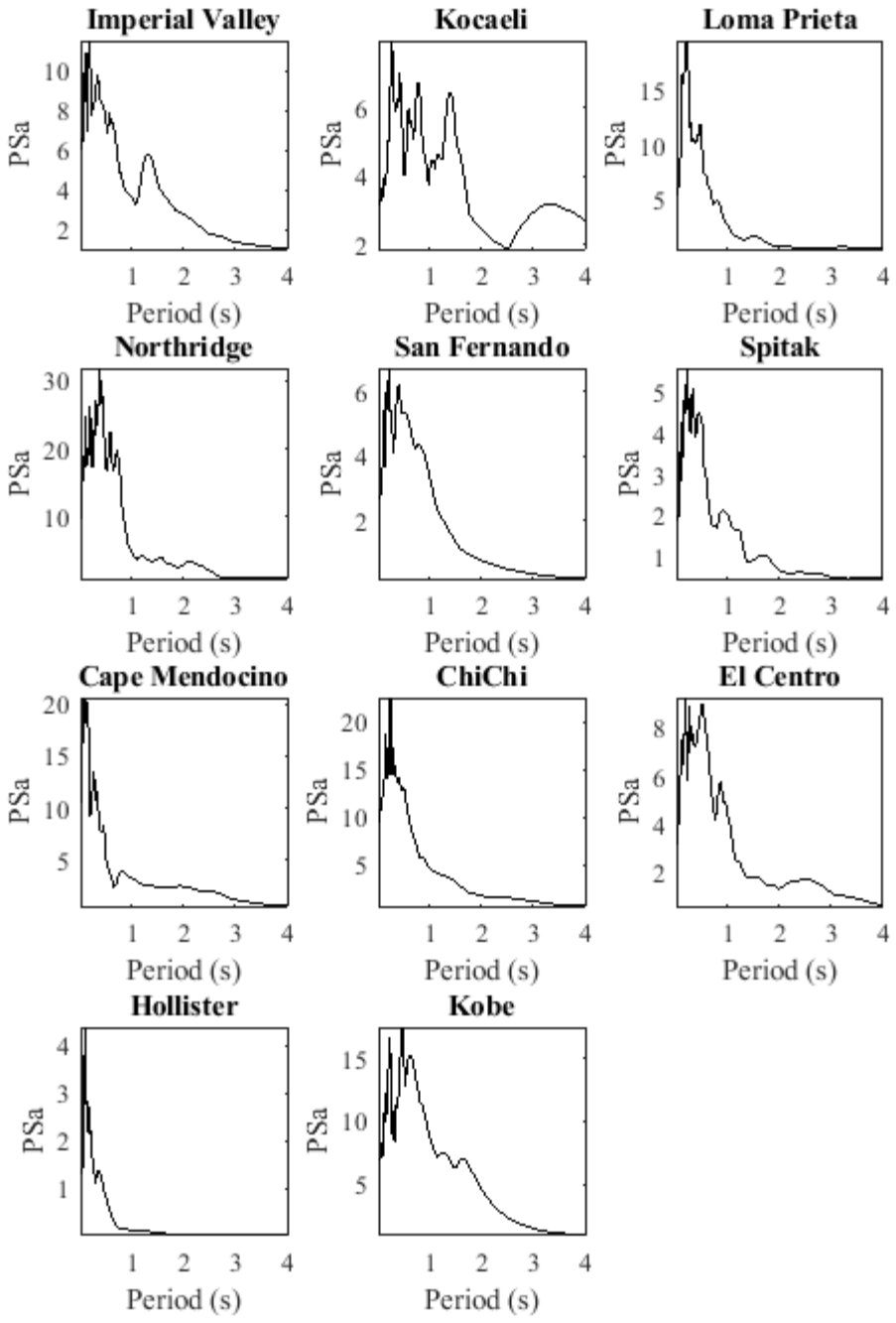
```
    axis tight
end
```



Plot pseudo-acceleration response spectra

```
Fig3 = figure('units', 'centimeters', 'Position', [0,0, 20/sqrt(2), 20]);
% Scan all subplots
for i=1:numel(eqmotions)
    subplot(4,3,i)
    plot(LERS{i}(:,1),LERS{i}(:,4),'k','LineWidth',1);
```

```
    set(gca,'FontName','Times New Roman')
    title(eqmotions{i},'FontName','Times New Roman')
    ylabel('PSa','FontName','Times New Roman')
    xlabel('Period (s)','FontName','Times New Roman')
    axis tight
end
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Comparison of elastic and constant ductility response spectra for mu=1

Compare the linear elastic acceleration response spectrum and the constant ductility response spectrum for mu=1 (degenerates to linear elastic). Also, compare with corresponding results of SeismoSignal software

Contents

Input

Earthquake motion

```
eqmotions={'Imperial Valley'}; % Imperial valley 1979
data=load([eqmotions{1},'.dat']);
t=data(:,1);
dt=t(2)-t(1);
xgtt=data(:,2);
```

Set the eigenperiod range for which the response spectra will be calculated.

```
Tspectra=(0.05:0.05:4)';
```

Set critical damping ratio of the response spectra to be calculated.

```
ksi=0.05;
```

First we use the CDS to calculate linear response

Set the target ductility

```
mu=1; % mu=1 equivalent to a linear SDoF
```

Extract constant ductility response spectra

```
sw='cds';
```

Calculation CDRS{i}=[S.Period,S.CDSd,S.CDSv,S.CDPSa,S.fyK,S.muK,S.iterK];

```
S1=OpenSeismoMatlab(dt,xgtt,sw,Tspectra,ksi,mu);
```

Next we use ES to calculate linear response

Extract linear elastic response spectra

```
sw='es';
```

Initialize LERS

```
LERS=cell(numel(eqmotions),1);
```

Calculation LERS{i}=[S.Period,S.Sd,S.PSv,S.PSa];

```
S2=OpenSeismoMatlab(dt,xgtt,sw,Tspectra,ksi);
```

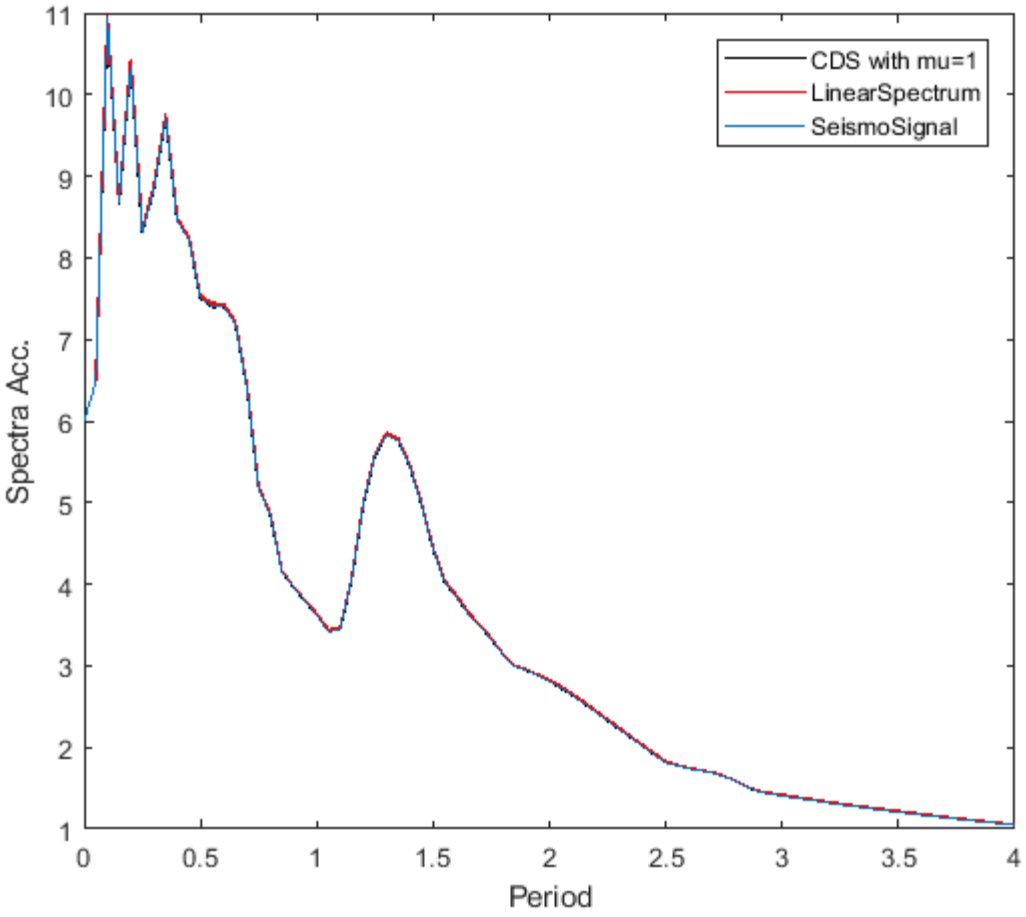
Results from SeismoSignal

```
fileID = fopen('SeismoSignal_Imperial Valley.txt');
for idx = 1:5
    fgetl(fileID);
end
C = textscan(fileID,repmat('%f',1,12));
fclose(fileID);
```

Output

Plot spectral acceleration response spectra

```
plot(S1.Period,S1.CDSa,'k','LineWidth',1);
hold on
plot(S2.Period,S2.Sa,'r','LineWidth',1);
plot(C{1},C{2})
legend('CDS with mu=1','LinearSpectrum','SeismoSignal');
xlabel('Period');
ylabel('Spectra Acc.')
```



Published with MATLAB® R2017b

Comparison of constant ductility response spectra of RSN1044 for mu=2

Contents

Input

Earthquake motions

```
GMFileLoc = 'input\';
```

A text file containing the lists of GM names

```
RecordList = ['input\Record_List.txt'];
```

Read E.Q. records name from the specifid list

```
fid = fopen (RecordList,'r');
records = textscan(fid, '%s'); % length = num of records*1, cell
fclose (fid);
```

Read E.Q. records name from the specifid list Return variables include record names, dt, num of points, PGA, Acc TH, Spectral Acc The accelration-related variables (PGA, Sa, Acc TH) are in unit of g; disp-related in m If you do not specify scaling factors at this step, the default value = 1 for each GM

```
[RecordName_all, dt, numPoint_all, PGA_all, xgtt] =...
    fn_ParseAT2File2(GMFileLoc, records);
```

Set the eigenperiod range for which the response spectra will be calculated.

```
Tspectra=(0.05:0.05:4)';
```

Set critical damping ratio of the response spectra to be calculated.

```
ksi=0.02;
```

Results from OpenSeismoMatlab

Set the target ductility

```
mu=2; % mu=1 equivalent to a linear SDoF
```

Extract constant ductility response spectra

```
sw='cds';
```

Calculation CDRS{i}=[S.Period,S.CDSd,S.CDSv,S.CDPSa,S.fyK,S.muK,S.iterK];

```
S1=OpenSeismoMatlab(dt,xgtt{1},sw,Tspectra,ksi,mu);
```

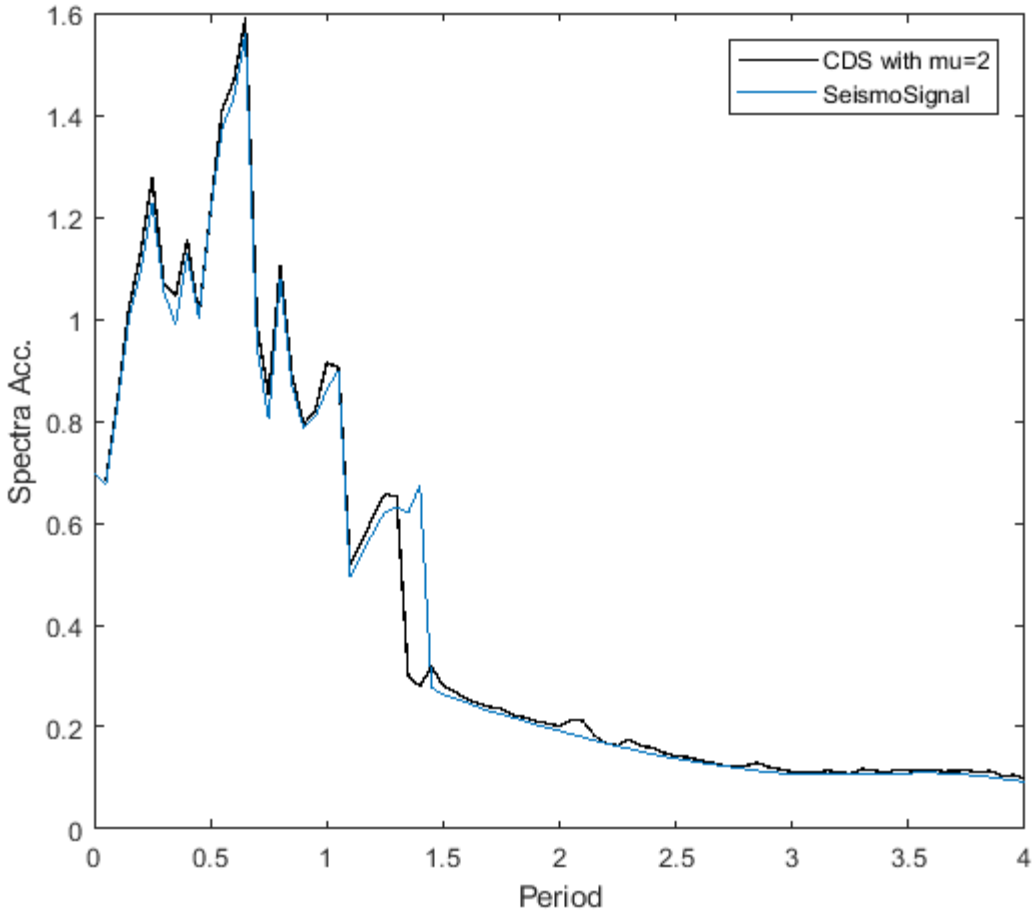
Results from SeismoSignal

```
fileID = fopen('input/RSN1044_Damping2.txt');
for idx = 1:5
    fgetl(fileID);
end
C = textscan(fileID,repmat('%f',1,15));
fclose(fileID);
```

Output

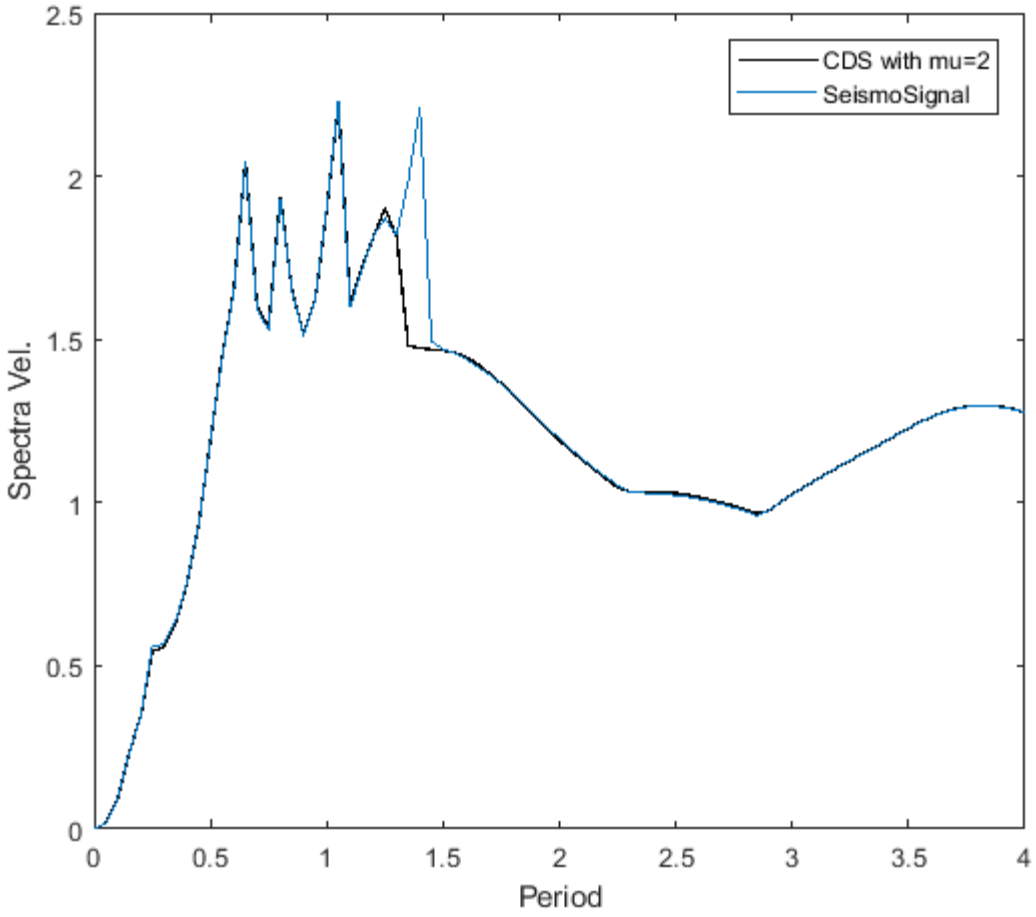
Plot spectral acceleration response spectra

```
figure()
plot(S1.Period,S1.CDSa/9.8,'k','LineWidth',1);
hold on
%plot(S2.Period,S2.Sa,'r','LineWidth',1);
plot(C{1},C{3})
legend('CDS with mu=2','SeismoSignal');
xlabel('Period');
ylabel('Spectra Acc.')
hold off
```



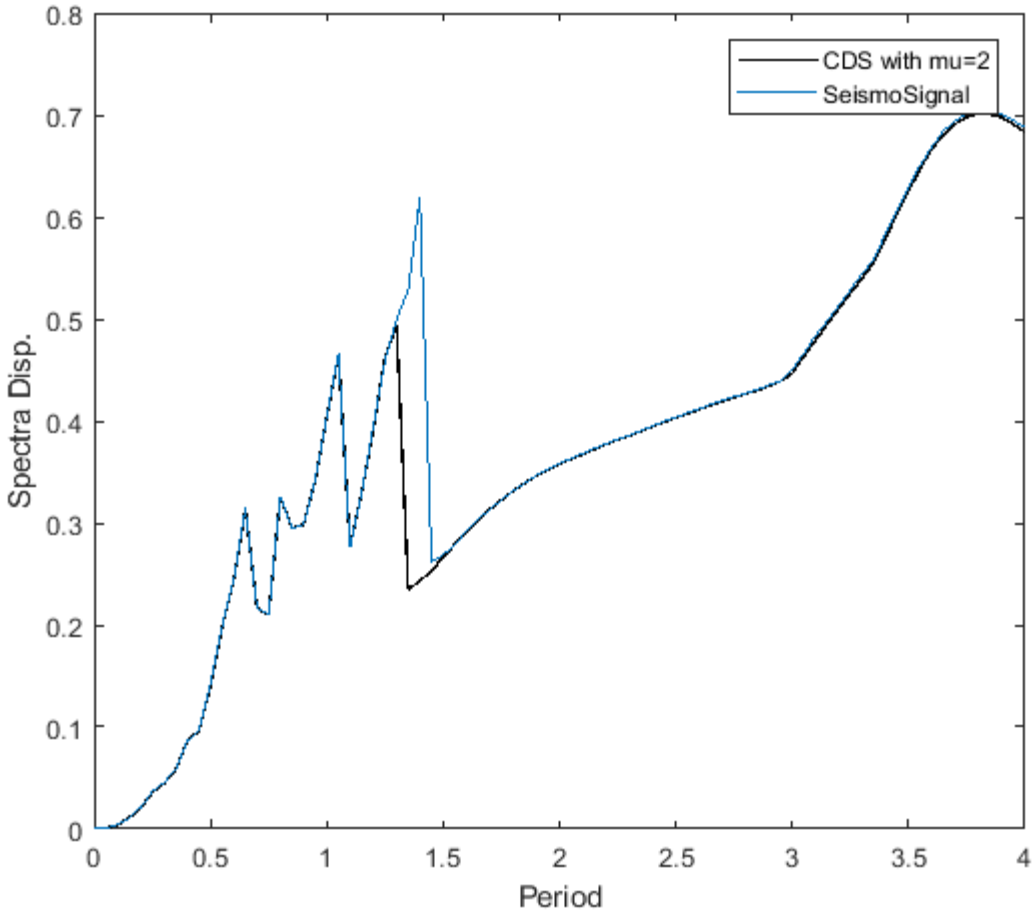
Plot spectral velocity response spectra

```
figure()
plot(S1.Period,S1.CDSv,'k','LineWidth',1);
hold on
%plot(S2.Period,S2.Sa,'r','LineWidth',1);
plot(C{1},C{7}/100)
legend('CDS with mu=2','SeismoSignal');
xlabel('Period');
ylabel('Spectra Vel.')
hold off
```



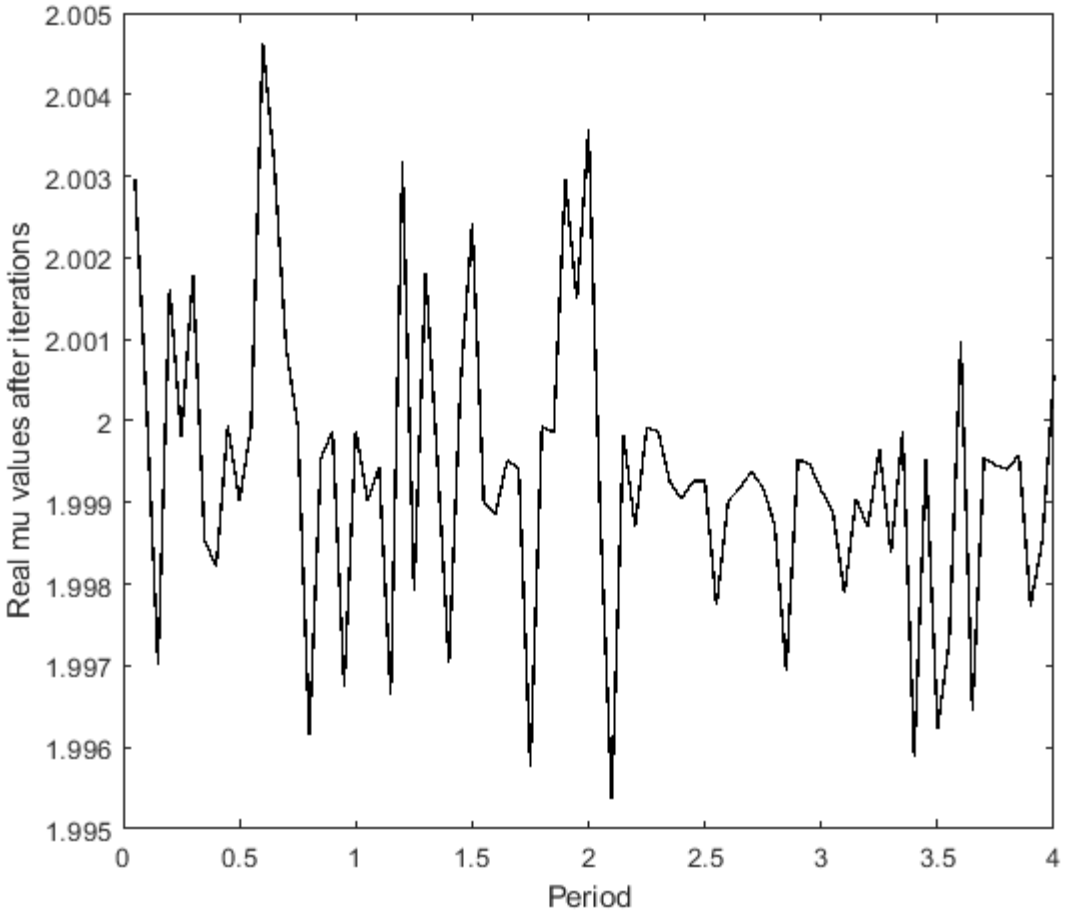
Plot spectral displacement response spectra

```
figure()
plot(S1.Period,S1.CDSd,'k','LineWidth',1);
hold on
%plot(S2.Period,S2.Sa,'r','LineWidth',1);
plot(C{1},C{11}/100)
legend('CDS with mu=2','SeismoSignal');
xlabel('Period');
ylabel('Spectra Disp.')
hold off
```



Plot the achieved ductility for each eigenperiod. It should be equal or close to the target ductility mu=2

```
figure()
plot(S1.Period,S1.muK,'k','LineWidth',1);
xlabel('Period');
ylabel('Real mu values after iterations')
```



Published with MATLAB® R2017b

Verify the dynamic response history analysis of OpenSeismoMatlab

Calculate linear dynamic response of a MDOF shear building

Contents

Reference

Chopra, A. K. (2020). Dynamics of structures, Theory and Applications to Earthquake Engineering, 5th edition. Prenctice Hall.

Description

The example 13.2.6 (Example: Five-Story Shear Frame) of the above reference is solved in this example. Consider the five-story shear frame of Fig.12.8.1 of the above reference, subjected to the El Centro ground motion. The lumped masses are equal to 45 Mg at each floor, the lateral stiffness of each story is 54.82 kN/cm and the height of each story is 4 m. The damping ratio for all natural modes is 0.05.

Load earthquake data

Earthquake acceleration time history of the El Centro earthquake will be used (El Centro, 1940, El Centro Terminal Substation Building)

```
fid=fopen('elcentro.dat','r');
text=textscan(fid,'%f %f');
fclose(fid);
t=text{1,1};
dt=t(2)-t(1);
xgtt=text{1,2};
```

Setup parameters for DRHA function

Set the storey height of the structure in m.

```
h=4;
```

Set the number of degrees of freedom of the structure, which is equal to the number of its storeys.

```
nDOFs=5;
```

Set the lateral stiffness of each storey in N/m.

```
k=5.482e6;
```

Set the lumped mass at each floor in kg.

```
m=45e3;
```

Calculation of structural properties

Calculate the stiffness matrix of the structure in N/m.

```
K=k*ones(nDOFs,1);
```

Calculate the mass matrix of the structure.

```
M=m*ones(nDOFs,1);
```

Critical damping ratio

```
ksi=0.05;
```

Initial displacement

```
u0=zeros(nDOFs,1);
```

Initial velocity

```
ut0=zeros(nDOFs,1);
```

Algorithm to be used for the time integration

```
AlgID='U0-V0-Opt';
```

Minimum absolute value of the eigenvalues of the amplification matrix

```
rinf=1;
```

Calculate dynamic response

Apply DRHA

```
[U,V,A,f,Es,Ed] = DRHA(K,M,dt,xgtt,ksi,u0,ut0,AlgID,rinf);
```

Base shear time history

```
FBeig=sum(f,1);
```

5th storey shear time history (5th DOF)

```
Feig=f(1,:);
```

Roof displacement time history (5th DOF)

```
Ueig=U(1,:);
```

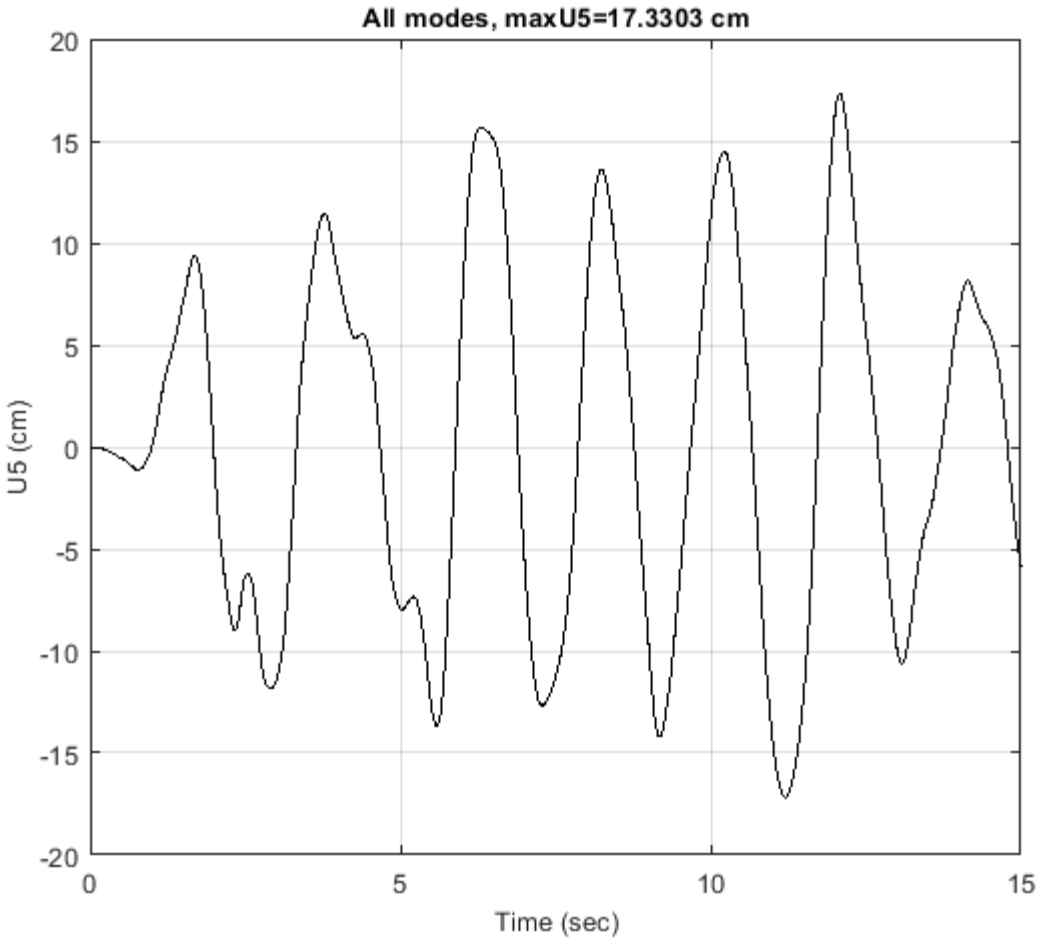
Base moment time history

```
MBeig=sum(f.*repmat((5*h:(-h):h)',1,size(f,2)),1);
```

Roof displacement time history

Plot the roof displacement time history. Convert displacements from m to cm. Verify with Figure 13.2.8 (left) of the above reference.

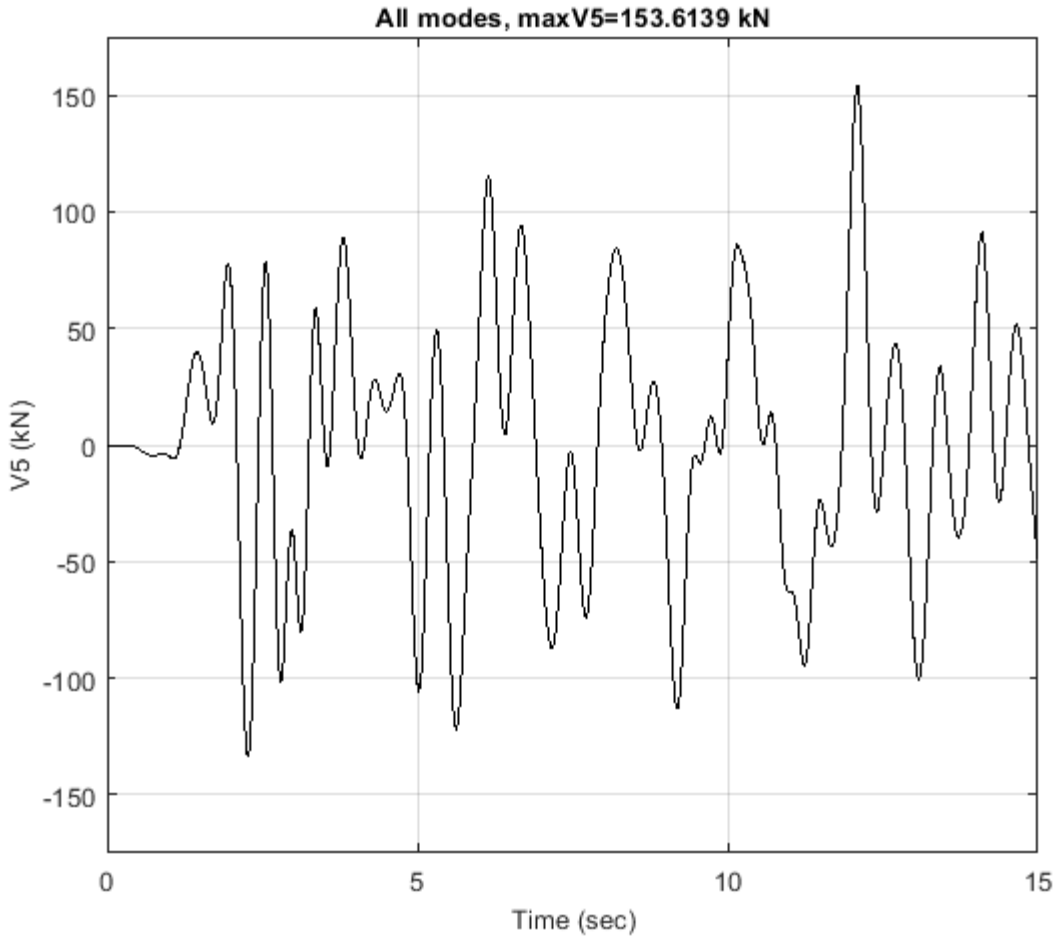
```
figure();
plot(t,100*Ueig,'LineWidth',1.,'Marker','.',...
    'MarkerSize',1,'Color',[0 0 0],'markeredgecolor','k')
grid on
xlim([0,15])
ylim([-20,20])
xlabel('Time (sec)','FontSize',10);
ylabel('U5 (cm)','FontSize',10);
title(['All modes, maxU5=',num2str(max(abs(100*Ueig))),' cm'],...
    'FontSize',10)
```



Fifth-story shear time history

Plot the fifth-story shear time history. Convert forces from N to kN. Verify with Figure 13.2.7 (right) of the above reference.

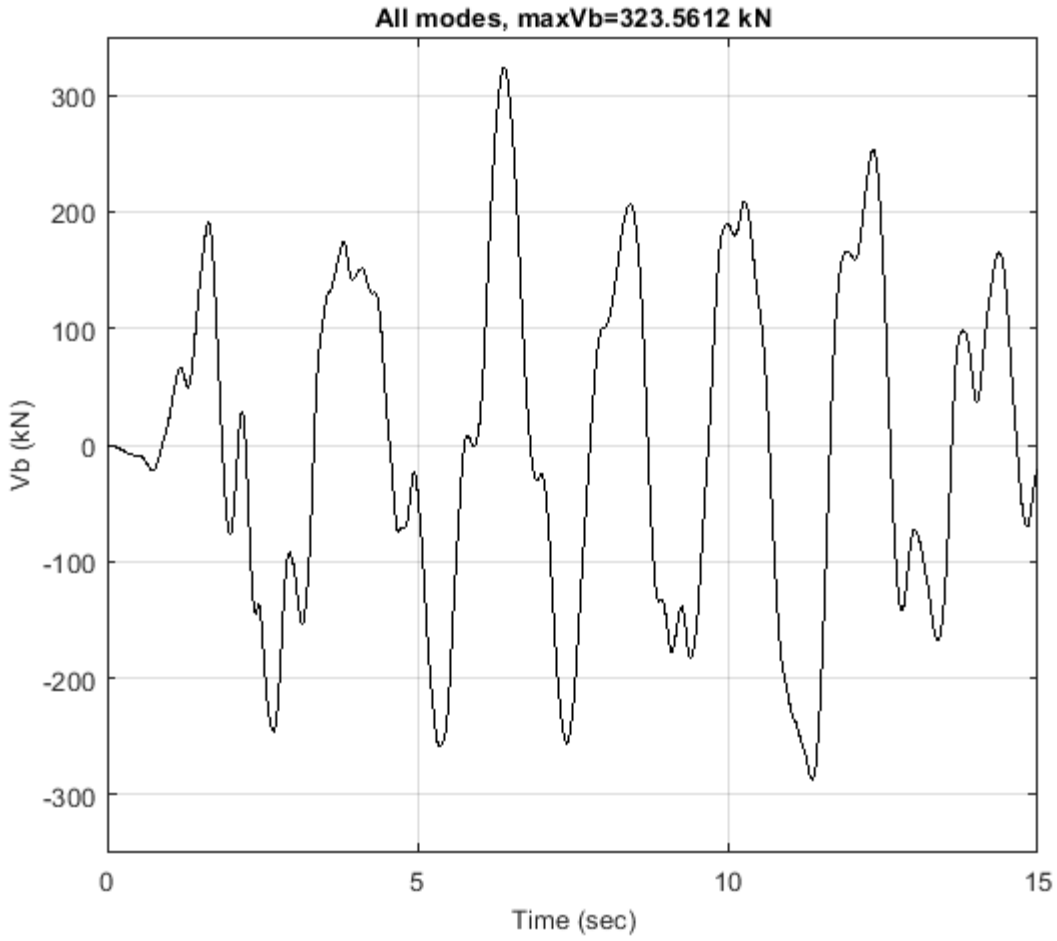
```
figure();
plot(t,Feig/1e3,'LineWidth',1.,'Marker','.',...
    'MarkerSize',1,'Color',[0 0 0],'markeredgecolor','k')
grid on
xlim([0,15])
ylim([-175,175])
xlabel('Time (sec)','FontSize',10);
ylabel('V5 (kN)','FontSize',10);
title(['All modes, maxV5=',num2str(max(abs(Feig/1e3))),' kN'],...
    'FontSize',10)
```



Base shear time history

Plot the base shear time history. Convert forces from N to kN. Verify with Figure 13.2.7 (left) of the above reference

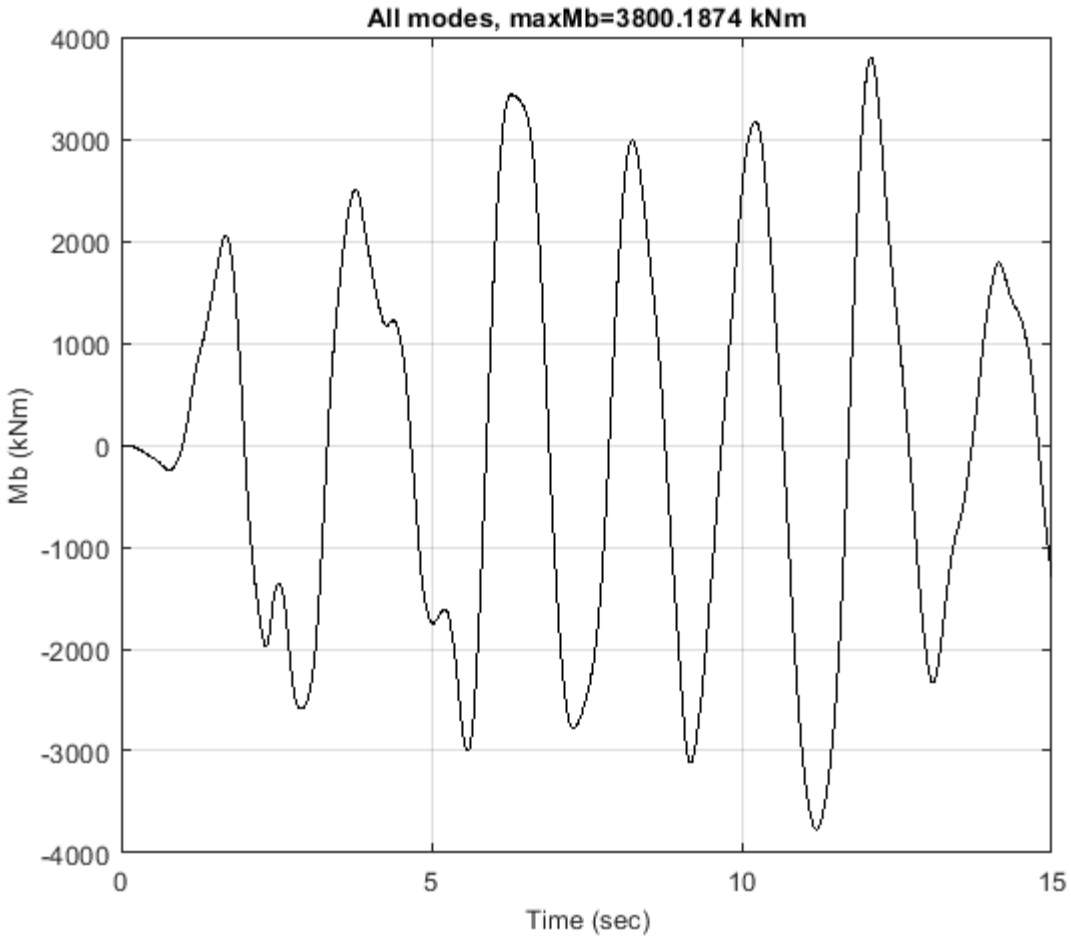
```
figure();
plot(t,FBeig/1e3,'LineWidth',1.,'Marker','.',...
    'MarkerSize',1,'Color',[0 0 0],'markeredgecolor','k')
grid on
xlim([0,15])
ylim([-350,350])
xlabel('Time (sec)','FontSize',10);
ylabel('Vb (kN)','FontSize',10);
title(['All modes, maxVb=',num2str(max(abs(FBeig/1e3))),' kN'],...
    'FontSize',10)
```



Base moment time history

Plot the base moment time history. Convert moments from Nm to kNm. Verify with Figure 13.2.8 (right) of the above reference

```
figure();
plot(t,MBeig/1e3,'LineWidth',1.,'Marker','.',...
    'MarkerSize',1,'Color',[0 0 0],'markeredgecolor','k')
grid on
xlim([0,15])
ylim([-4000,4000])
xlabel('Time (sec)','FontSize',10);
ylabel('Mb (kNm)','FontSize',10);
title(['All modes, maxMb=',num2str(max(abs(MBeig/1e3))),' kNm'],...
    'FontSize',10)
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the high pass Butterworth filter of OpenSeismoMatlab

Contents

Reference

Boore, D. M. (2005). On pads and filters: Processing strong-motion data. Bulletin of the Seismological Society of America, 95(2), 745-750.

Description

Verify Figure 1 of the above reference, for the 1940 El Centro analog recording. The displacements and velocities from unfiltered and filtered accelerations are shown. Filtering is done by using two passes of a fourth-order high pass (i.e. frequencies lower than the cut-off frequency are attenuated) Butterworth filter with cut-off frequency as shown in Figure 1.

Earthquake motion

Load earthquake data

```
eqmotions={'Imperial_Valley_El_Centro_9_EW'};
data=load([eqmotions{1},'.dat']);
t=data(:,1);
dt=t(2)-t(1);
xgtt=data(:,2);
```

Apply high pass Butterworth filter

Switch

```
sw='butterworthhigh';
```

Order of Butterworth filter

```
bOrder=4;
```

Cut-off frequency

```
flc=0.1;
```

Apply OpenSeismoMatlab

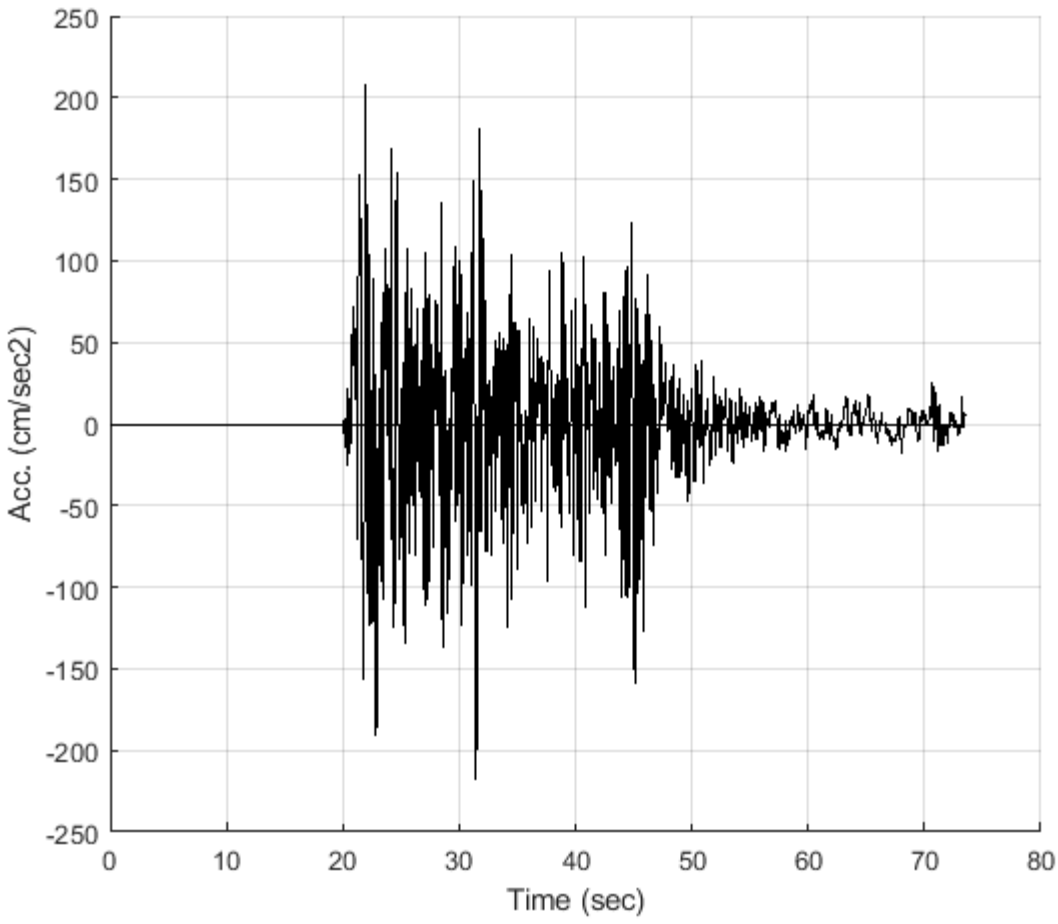
```
S1=OpenSeismoMatlab(dt,xgtt,sw,bOrder,flc);
```

Filtered acceleration

```
cxgtt=S1.acc;
```

Plot the time history of the initial ground motion

```
% Initialize figure
figure()
hold on
plot(t,zeros(size(t)),'k','LineWidth',1)
% Plot the acceleration time history of the initial ground motion
plot(t,xgtt,'k','LineWidth',1)
% Finalize figure
hold off
grid on
xlabel('Time (sec)')
ylabel('Acc. (cm/sec2)')
```



Obtain displacement and velocity time histories

Switch

```
sw='timehist';
```

Do not use baseline correction

```
baselineSw=false;
```

Apply OpenSeismoMatlab to the initial ground motion

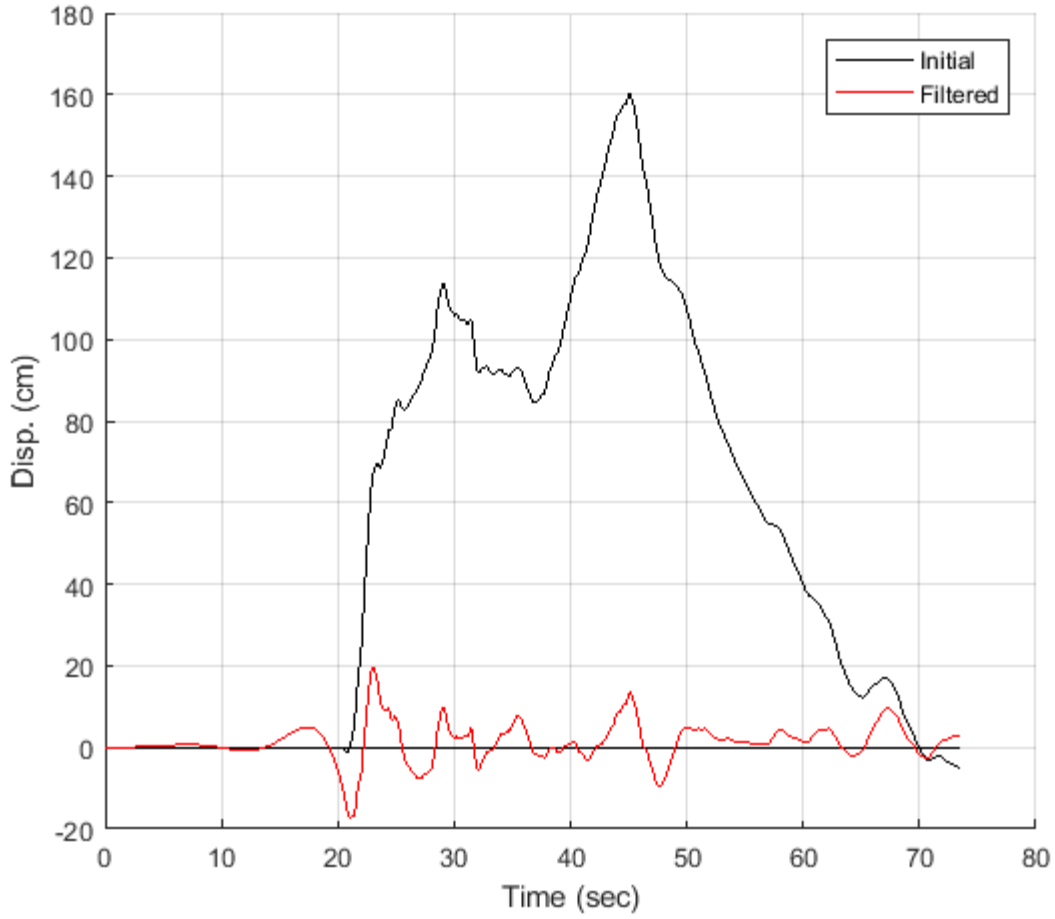
```
S2=OpenSeismoMatlab(dt,xgtt,sw,baselineSw);
```

Apply OpenSeismoMatlab to the filtered ground motion

```
S3=OpenSeismoMatlab(dt,cxgtt,sw,baselineSw);
```

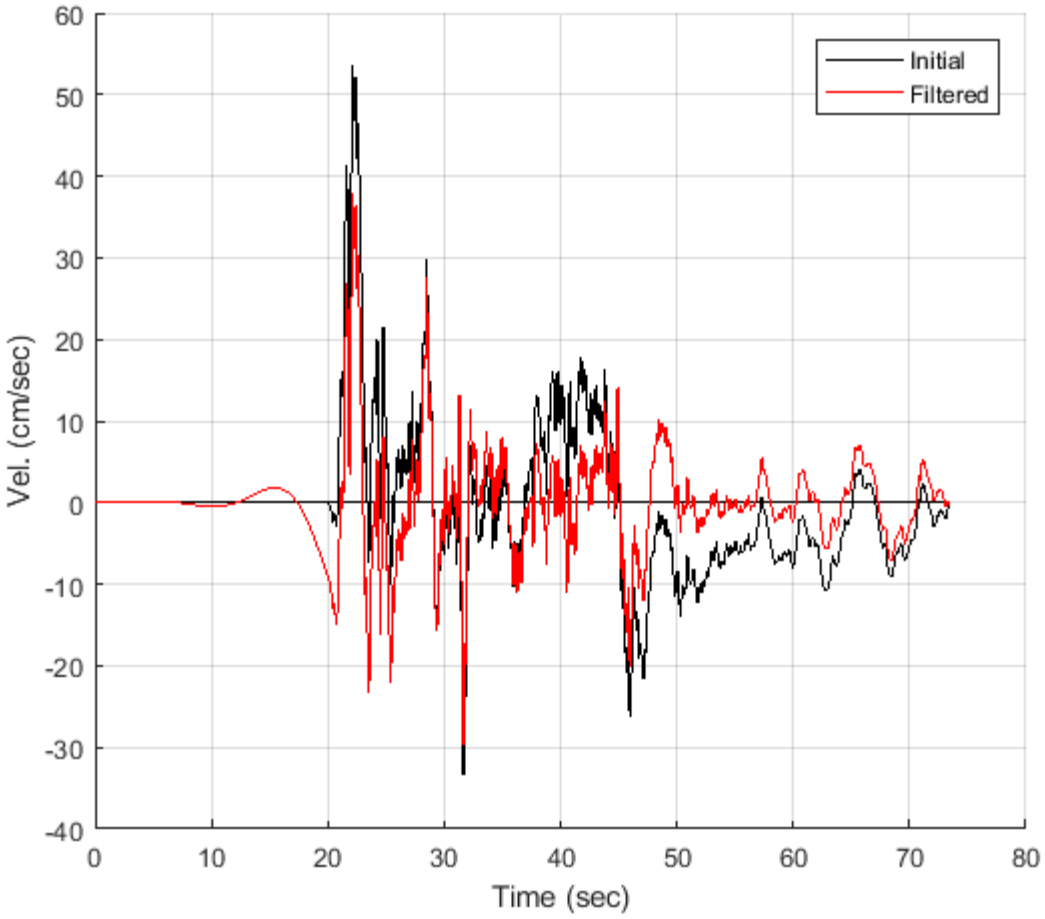
Plot the displacement time histories

```
% Initialize figure
figure()
hold on
plot(S3.time,zeros(size(S3.time)),'k','LineWidth',1)
% Plot the displacement time history of the initial ground motion
p1=plot(S2.time,S2.disp,'k','LineWidth',1);
% Plot the displacement time history of the filtered ground motion
p2=plot(S3.time,S3.disp,'r','LineWidth',1);
% Finalize figure
hold off
grid on
legend([p1,p2],{'Initial','Filtered'})
xlabel('Time (sec)')
ylabel('Disp. (cm)')
```



Plot the velocity time histories

```
% Initialize figure
figure()
hold on
plot(S3.time,zeros(size(S3.time)),'k','LineWidth',1)
% Plot the velocity time history of the initial ground motion
p1=plot(S2.time,S2.vel,'k','LineWidth',1);
% Plot the velocity time history of the filtered ground motion
p2=plot(S3.time,S3.vel,'r','LineWidth',1);
% Finalize figure
hold off
grid on
legend([p1,p2],{'Initial','Filtered'})
xlabel('Time (sec)')
ylabel('Vel. (cm/sec)')
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the low- and high- pass Butterworth filter of OpenSeismoMatlab

Contents

Reference

Graizer, V. (2012, September). Effect of low-pass filtering and re-sampling on spectral and peak ground acceleration in strong-motion records. In Proceedings of the 15th World Conference of Earthquake Engineering, Lisbon, Portugal (pp. 24-28).

Description

Verify Figure 3.2 of the above reference, for the the MW 6.3 Christchurch, New Zealand earthquake at Heathcote Valley Primary School (HVSC) station, Up-component. The time histories, elastic response spectra and Fourier spectra from unfiltered and filtered accelerations are shown and compared. In the above reference, the ground motion was processed following the 1970s Caltech procedure, low-pass filtered and re-sampled to 50 samples/sec by the GeoNet New Zealand strong motion network. However in this example, Butterworth filter is applied and it gives similar results.

Earthquake motion

Load earthquake data

```
eqmotions={'Christchurch2011HVPS_UP'};
data=load([eqmotions{1},'.dat']);
t=data(:,1);
dt=t(2)-t(1);
xgtt=data(:,2);
xgtt=[zeros(10/dt,1);xgtt];
t=(0:numel(xgtt)-1)'*dt;
xgtt=xgtt/9.81;
```

Apply high pass Butterworth filter

Switch

```
sw='butterworthhigh';
```

Order of Butterworth filter

```
bOrder=4;
```

Cut-off frequency

```
flc=0.1;
```

Apply OpenSeismoMatlab

```
S1=OpenSeismoMatlab(dt,xgtt,sw,bOrder,flc);
```

Filtered acceleration

```
cxgtt=S1.acc;
```

Apply low pass Butterworth filter

Switch

```
sw='butterworthlow';
```

Order of Butterworth filter

```
bOrder=4;
```

Cut-off frequency

```
fuc=25;
```

Apply OpenSeismoMatlab to high pass filtered filtered acceleration

```
S2=OpenSeismoMatlab(dt,cxgtt,sw,bOrder,fuc);
```

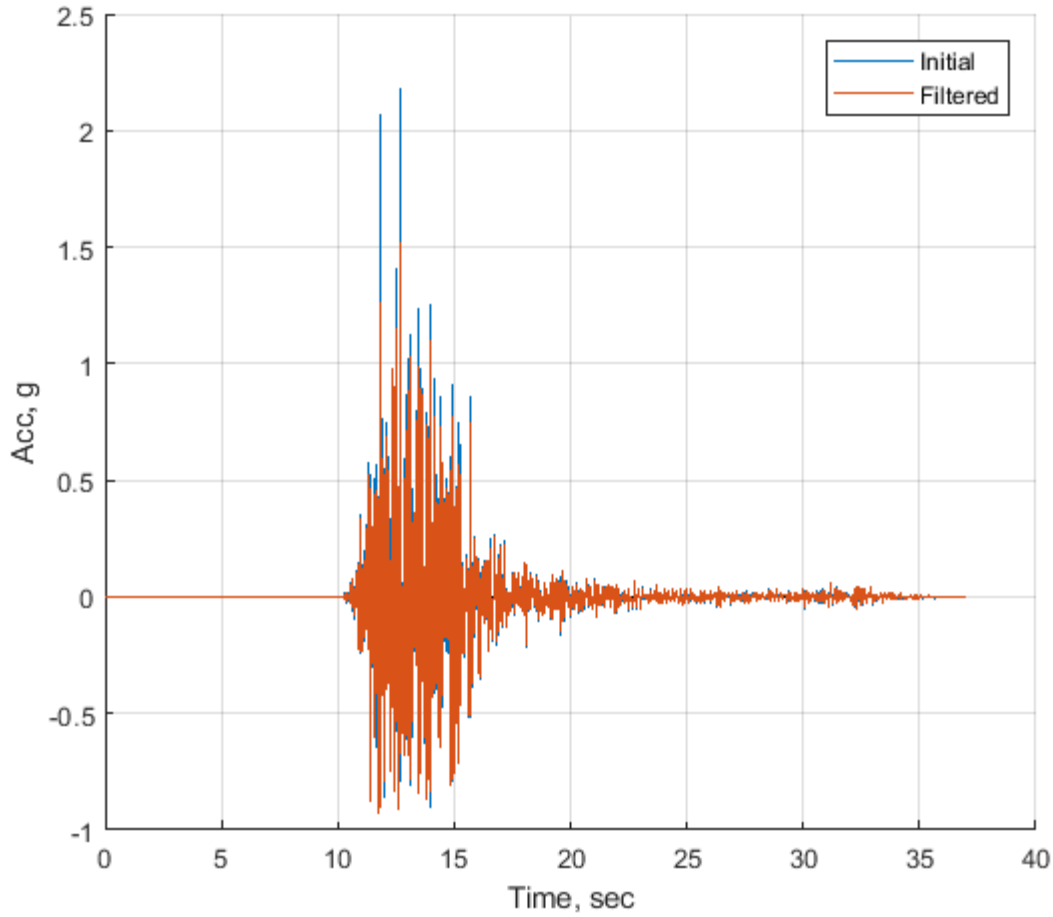
Filtered acceleration

```
cxgtt=S2.acc;
```

Plot the acceleration time histories

```
% Initialize figure
figure()
hold on
plot(t,zeros(size(t)),'k','LineWidth',1)
% Plot the acceleration time history of the initial ground motion
p1=plot(t,xgtt);
% Plot the acceleration time history of the bandpass filtered ground motion
```

```
p2=plot(t,cxgtt);
% Finalize figure
hold off
grid on
legend([p1,p2],{'Initial','Filtered'})
xlabel('Time, sec')
ylabel('Acc, g')
```



Calculate the Fourier spectra

Switch

```
sw='fs';
```

Apply OpenSeismoMatlab to the initial ground motion

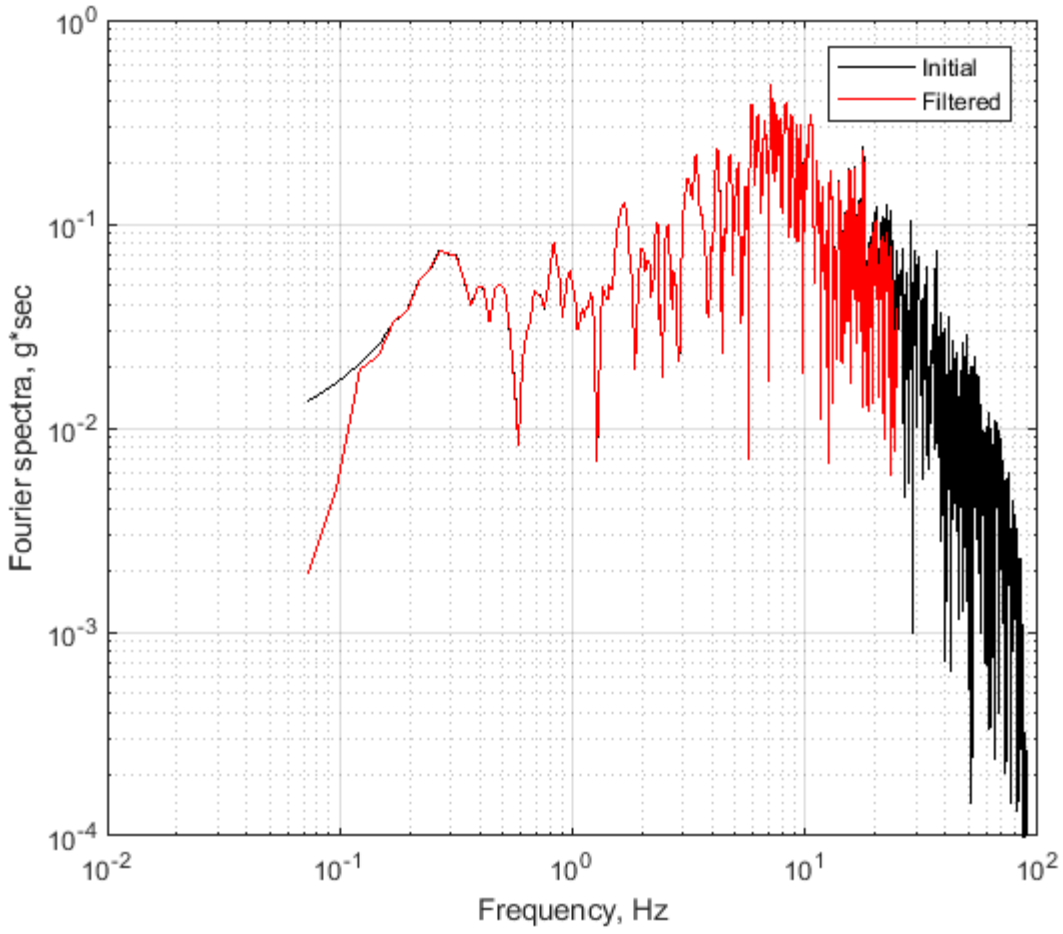
```
S3=OpenSeismoMatlab(dt,xgtt,sw);
```

Apply OpenSeismoMatlab to the filtered ground motion

```
S4=OpenSeismoMatlab(dt,cxgtt,sw);
```

Plot the Fourier spectra

```
% Initialize figure
figure()
loglog(1,1,'w')
hold on
% Plot the Fourier spectrum of the initial ground motion in logarithmic
% scale for frequencies larger than 0.05 Hz
ind10=S4.freq>=0.05;
p1=loglog(S3.freq(ind10),S3.FAS(ind10),'k','LineWidth',1);
% Plot the Fourier spectrum of the filtered ground motion in logarithmic
% scale for frequencies larger than 0.05 Hz and lower than fuc
ind11=(S4.freq<=fuc)& (S4.freq>=0.05);
p2=loglog(S4.freq(ind11),S4.FAS(ind11),'r','LineWidth',1);
% Finalize figure
hold off
grid on
ylim([1e-4,1])
legend([p1,p2],{'Initial','Filtered'})
xlabel('Frequency, Hz')
ylabel('Fourier spectra, g*sec')
```



Calculate the acceleration response spectra

Switch

```
sw='es';
```

Critical damping ratio

```
ksi=0.05;
% Period range for which the response spectrum is queried
T=logspace(1,-2,100);
```

Apply OpenSeismoMatlab to the initial ground motion

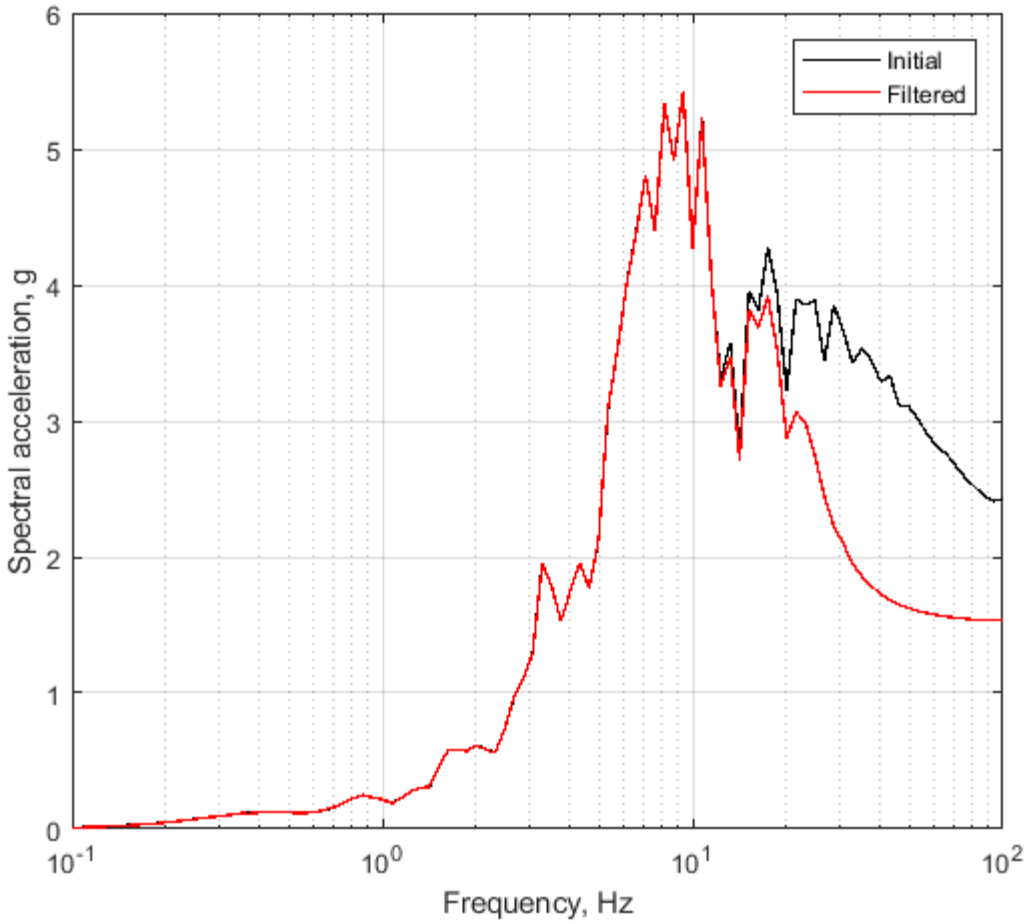
```
S5=OpenSeismoMatlab(dt,xgtt,sw,T,ksi);
```

Apply OpenSeismoMatlab to the filtered ground motion

```
S6=OpenSeismoMatlab(dt,cxgtt,sw,T,ksi);
```

Plot the acceleration response spectra

```
% Initialize figure
figure()
semilogx(1,1,'w')
hold on
% Plot the acceleration response spectrum of the initial ground motion in
% logarithmic scale
p1=semilogx(1./S5.Period,S5.Sa,'k','LineWidth',1);
% Plot the acceleration response spectrum of the filtered ground motion in
% logarithmic scale
p2=semilogx(1./S6.Period,S6.Sa,'r','LineWidth',1);
% Finalize figure
hold off
grid on
legend([p1,p2],{'Initial','Filtered'})
xlabel('Frequency, Hz')
ylabel('Spectral acceleration, g')
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the Fourier amplitude spectrum of OpenSeismoMatlab

Contents

Reference

Analyses of strong motion earthquake accelerograms, Volume IV - Fourier Amplitude Spectra, Part H - Accelerograms IIH115 through IIH126, California Institute of Technology, Earthquake Engineering Research Laboratory, Report No. EERl 74-100, 1974.

Description

Verify the Fourier amplitude spectrum at page 12 of the above reference for the San Fernando earthquake, Feb 9, 1971, 0600 PST, IVH115 71.024.0 15250 Ventura BLVD., basement, Los Angeles, Cal. Component N11E.

Earthquake motion

Load earthquake data

```
eqmotions={'SanFernando1971VenturaBlvdBasement15250LosAngelesCalN11E'};
data=load([eqmotions{1},'.dat']);
t=data(:,1);
dt=t(2)-t(1);
xgtt=100*data(:,2);
```

Calculate the Fourier amplitude spectrum

Switch

```
sw='fs';
```

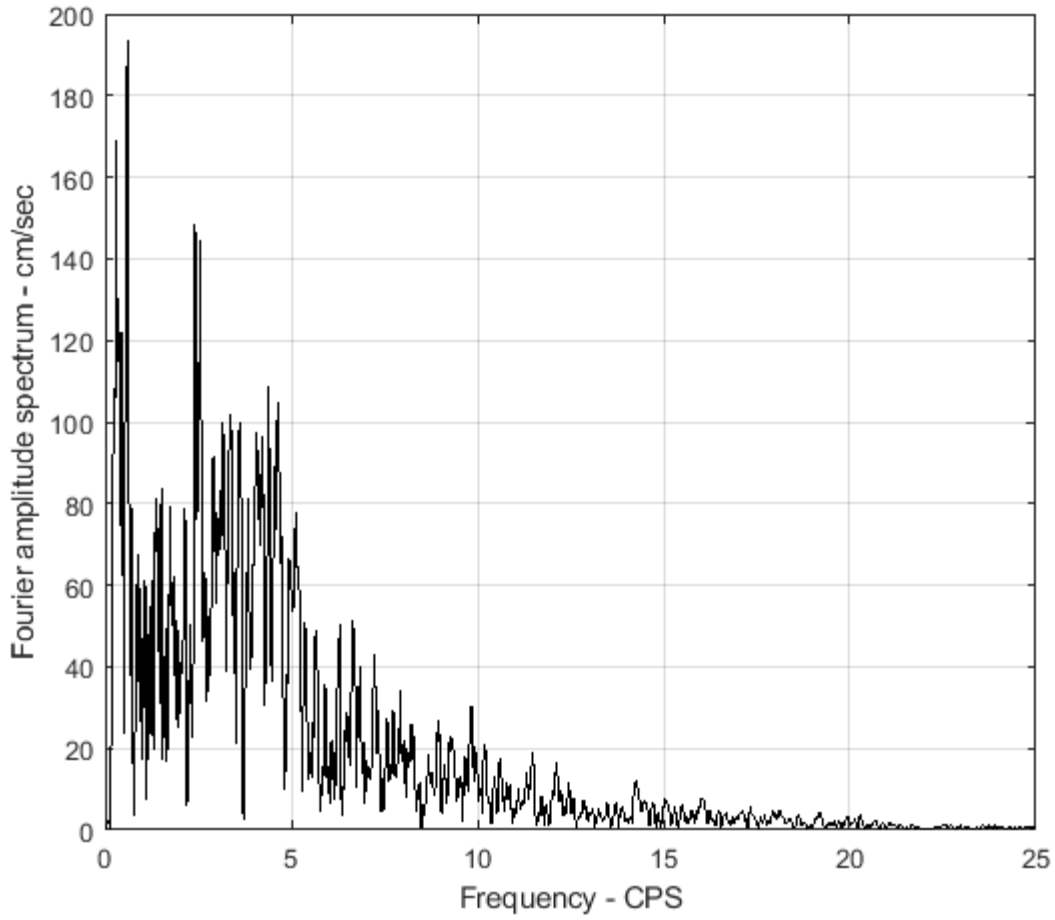
Apply OpenSeismoMatlab

```
S1=OpenSeismoMatlab(dt,xgtt,sw);
```

Plot the Fourier amplitude spectrum

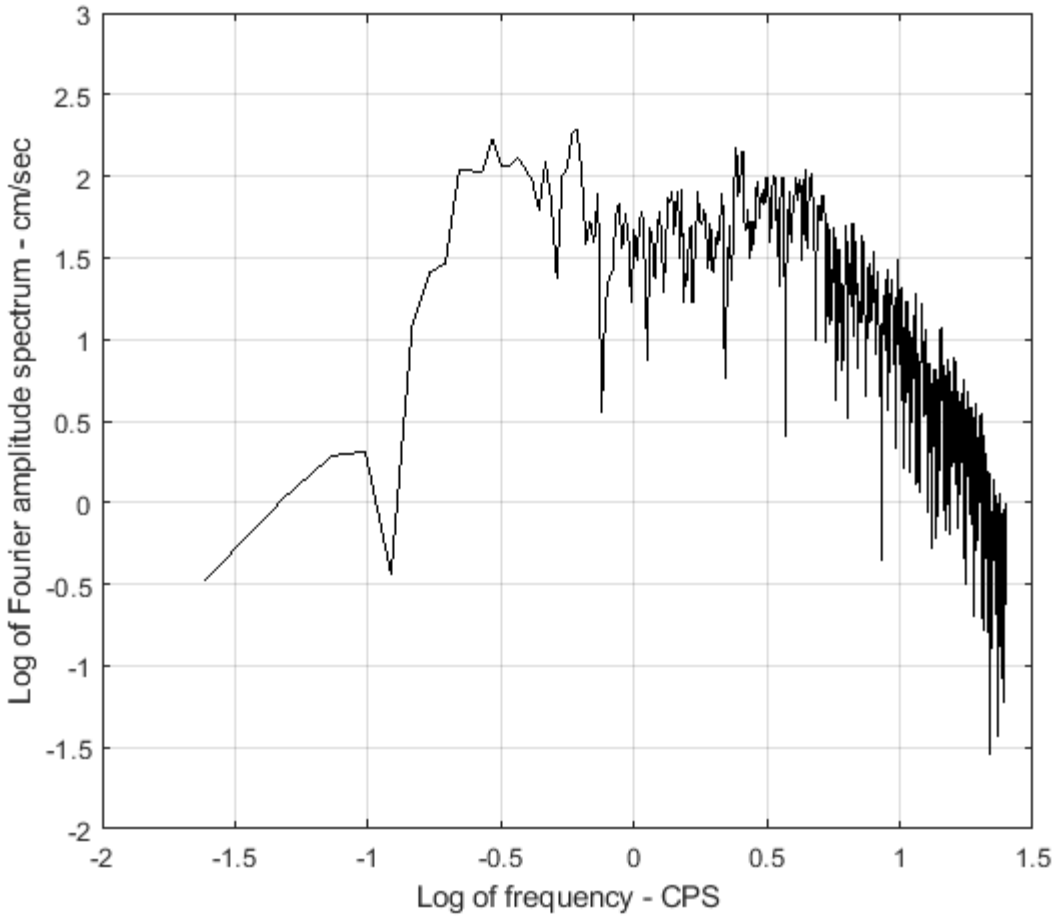
```
% Initialize figure
figure()
% Plot the Fourier amplitude spectrum on page 12 of the above reference
plot(S1.freq,S1.FAS,'k','LineWidth',1)
% Finalize figure
grid on
```

```
xlabel('Frequency - CPS')
ylabel('Fourier amplitude spectrum - cm/sec')
ylim([0,200])
xlim([0,25])
```



Plot the Fourier amplitude spectrum in logarithmic scale

```
% Initialize figure
figure()
% Plot the Fourier amplitude spectrum on page 13 of the above reference
plot(log10(S1.freq),log10(S1.FAS),'k','LineWidth',1)
% Finalize figure
grid on
xlabel('Log of frequency - CPS')
ylabel('Log of Fourier amplitude spectrum - cm/sec')
ylim([-2,3])
xlim([-2,1.5])
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the Fourier amplitude spectrum of OpenSeismoMatlab

Contents

Reference

Analyses of strong motion earthquake accelerograms, Volume IV - Fourier Amplitude Spectra, Part H - Accelerograms IIH115 through IIH126, California Institute of Technology, Earthquake Engineering Research Laboratory, Report No. EERl 74-100, 1974.

Description

Verify the Fourier amplitude spectrum at page 14 of the above reference for the San Fernando earthquake, Feb 9, 1971, 0600 PST, IVH115 71.024.0 15250 Ventura BLVD., basement, Los Angeles, Cal. Component N79W.

Earthquake motion

Load earthquake data

```
eqmotions={'SanFernando1971VenturaBlvdBasement15250LosAngelesCalN79W'};
data=load([eqmotions{1},'.dat']);
t=data(:,1);
dt=t(2)-t(1);
xgtt=100*data(:,2);
```

Calculate the Fourier amplitude spectrum

Switch

```
sw='fs';
```

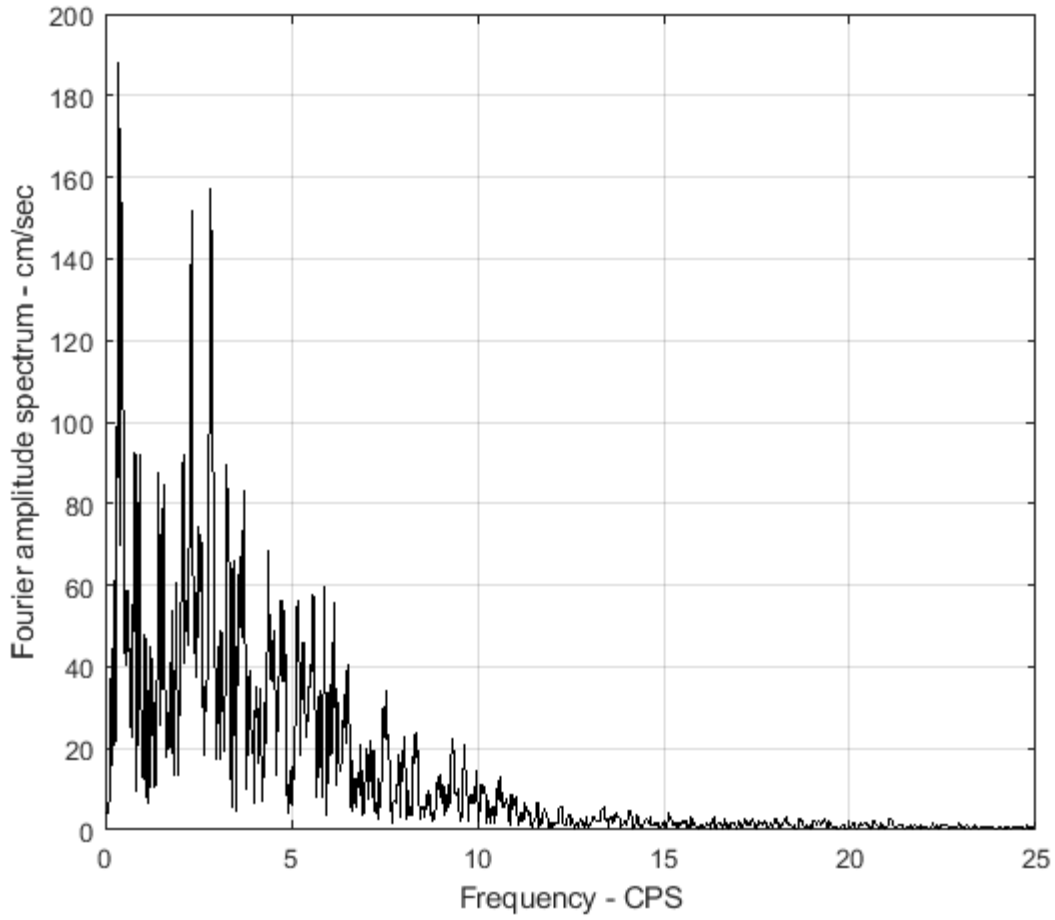
Apply OpenSeismoMatlab

```
S1=OpenSeismoMatlab(dt,xgtt,sw);
```

Plot the Fourier amplitude spectrum

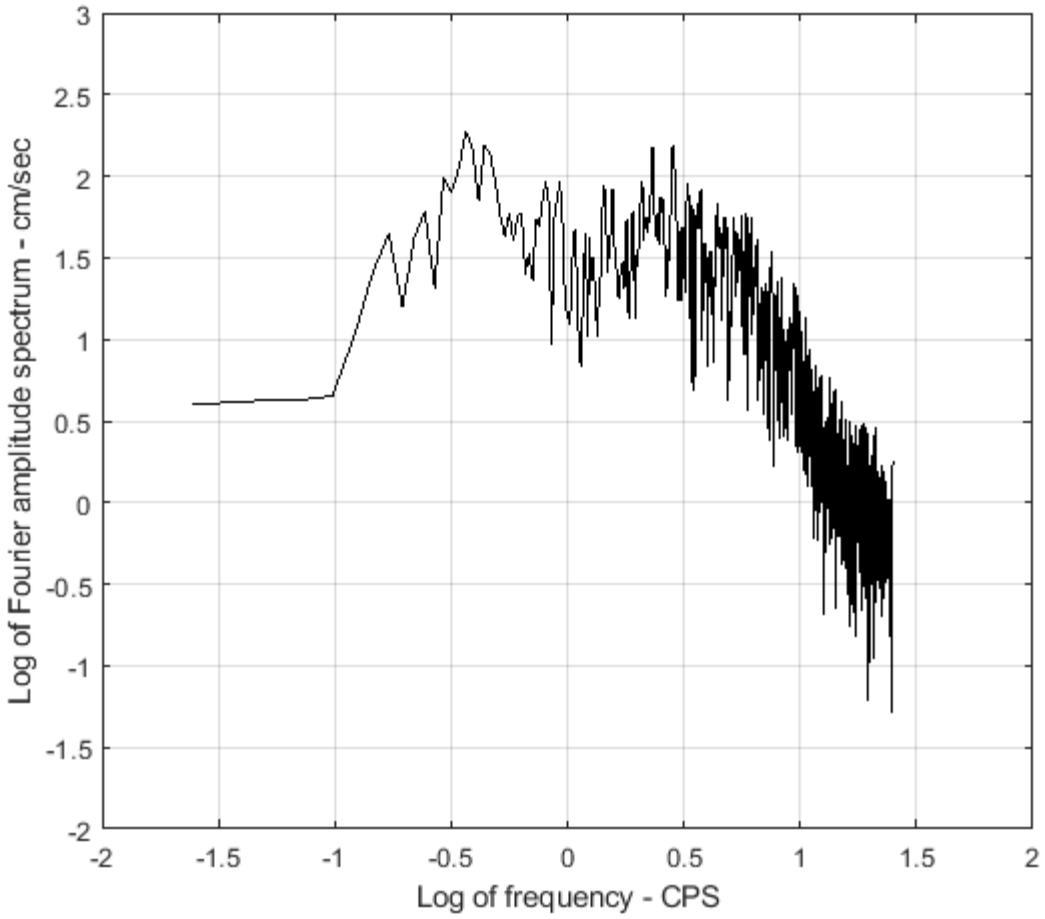
```
% Initialize figure
figure()
% Plot the Fourier amplitude spectrum on page 14 of the above reference
plot(S1.freq,S1.FAS,'k','LineWidth',1)
% Finalize figure
grid on
```

```
xlabel('Frequency - CPS')
ylabel('Fourier amplitude spectrum - cm/sec')
ylim([0,200])
xlim([0,25])
```



Plot the Fourier amplitude spectrum in logarithmic scale

```
% Initialize figure
figure()
% Plot the Fourier amplitude spectrum on page 15 of the above reference
plot(log10(S1.freq),log10(S1.FAS),'k','LineWidth',1)
% Finalize figure
grid on
xlabel('Log of frequency - CPS')
ylabel('Log of Fourier amplitude spectrum - cm/sec')
ylim([-2,3])
xlim([-2,2])
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the incremental dynamic analysis of OpenSeismoMatlab

Contents

Reference

Mashayekhi, M., Harati, M., Darzi, A., & Estekanchi, H. E. (2020). Incorporation of strong motion duration in incremental-based seismic assessments. Engineering Structures, 223, 111144.

Description

Incremental dynamic analysis (IDA) is performed for a non-degrading SDOF model with eigenperiod T=1 sec. The employed hysteretic model is a bilinear elastoplastic model used for non-degrading SDOF systems and is shown in Figure 17(a) of the above reference. An IDA analysis is performed with a ground motion the spectral acceleration of which resembles the red line of Figure 14 of the above reference, i.e. the ground motion must have Sa(1 sec)=0.382g (which is the Intensity Measure - IM) and the duration D_5_75 must be roughly equal to 8.3 sec. An acceleration time history with such characteristics is shown in Figure 16(c) of the above reference. In this example, an arbitrary ground motion acceleration is loaded, which is then adjusted so that the resulting time history has Sa(1 sec)=0.382g and D_5_75=8.3 sec. The adjusted time history is plotted in this example and can be compared to Figure 16(c) of the above reference. Based on the above problem statement, the median response curve of Figure 18(a) of the above reference is verified.

Earthquake motion

Load earthquake data

```
eqmotions={'LomaPrietaHallsValley90'};
data=load([eqmotions{1},'.dat']);
t=data(:,1);
dt=t(2)-t(1);
xgtt=data(:,2);
```

Adjust earthquake motion to have D_5_75=8.3sec

Switch

```
sw='arias';
```

Apply OpenSeismoMatlab

```
S1=OpenSeismoMatlab(dt,xgtt,sw);
```

Duration D_5_75 of the initially loaded motion

```
S1.Td_5_75
```

```
ans =

   7.759999999999999
```

S.Td_5_75 must be roughly near 8.3 sec, as required in Mashayekhi et al. (2020) We manipulate the strong shaking part of the motion which corresponds to the significant duration so that S.Td_5_75 is increased to the desired value (8.3 sec)

```
id1=find(t==S1.t_5_75(1));
id2=find(t==S1.t_5_75(2));
xgtt(id1:id2)=0.8*xgtt(id1:id2);
```

Calculate duration D_5_75 of adjusted earthquake motion

Switch

```
sw='arias';
```

Apply OpenSeismoMatlab

```
S2=OpenSeismoMatlab(dt,xgtt,sw);
```

Duration D_5_75 of the adjusted motion

```
S2.Td_5_75
```

```
ans =

   8.359999999999999
```

Scale earthquake motion to have Sa(1 sec)=0.382g

Switch

```
sw='es';
```

Critical damping ratio

```
ksi=0.05;
```

Period where Sa=0.382g

```
T=1;
```

Apply OpenSeismoMatlab

```
S3=OpenSeismoMatlab(dt,xgtt,sw,T,ksi);
```

Spectral acceleration of the adjusted motion at 1 sec

```
S3.Sa
```

```
ans =

   1.286911370102642
```

Sa at 1 sec must be equal to 0.382g, so we scale the entire acceleration time history up to this level

```
scaleF=0.382*9.81/S3.Sa;
xgtt=xgtt*scaleF;
```

Calculate spectral acceleration of scaled earthquake motion

Switch

```
sw='es';
```

Critical damping ratio

```
ksi=0.05;
```

Period where Sa=0.382g

```
T=1;
```

Apply OpenSeismoMatlab

```
S4=OpenSeismoMatlab(dt,xgtt,sw,T,ksi);
```

Spectral acceleration of the adjusted motion at 1 sec

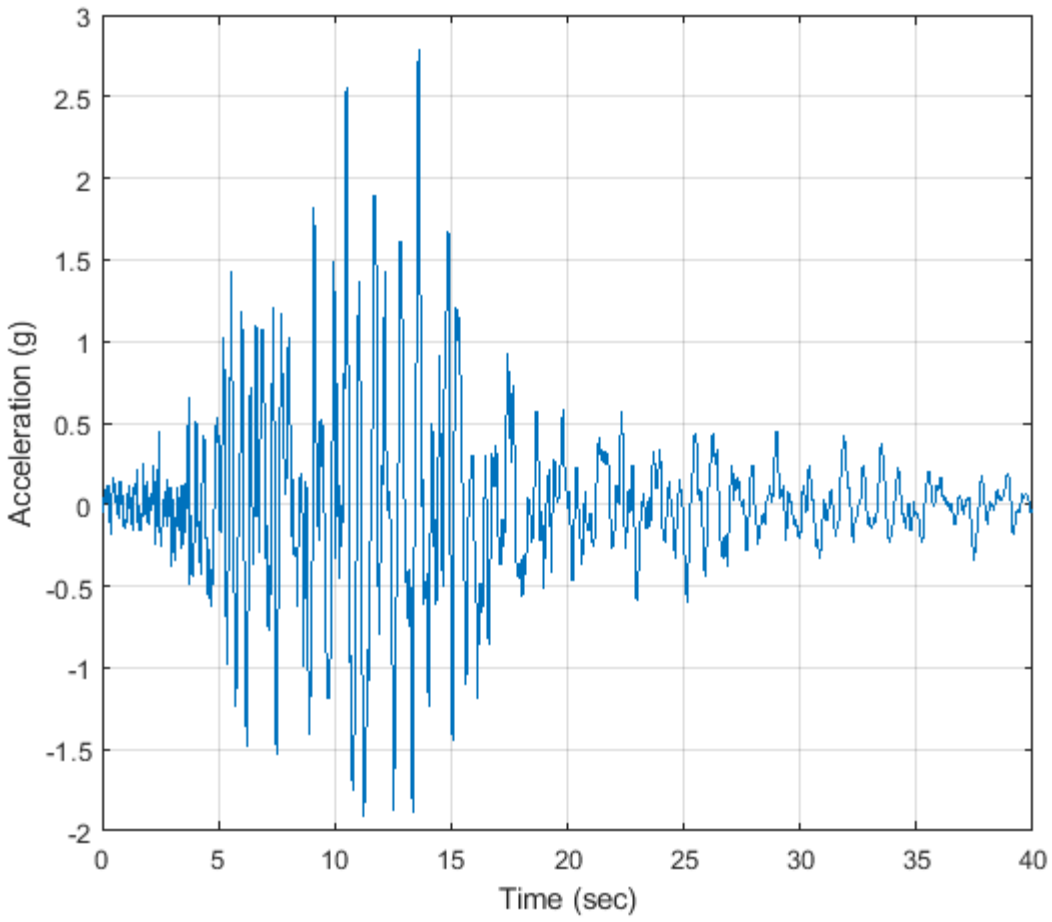
```
S4.Sa
```

```
ans =

   3.747419999999983
```

Plot the acceleration time history

```
% Initialize figure
figure()
% Plot the acceleration time history of the adjusted motion
plot(t,xgtt)
% Finalize figure
grid on
xlabel('Time (sec)')
ylabel('Acceleration (g)')
```



Perform IDA analysis

Switch

```
```

```
sw='ida';
```

Eigenperiod

```
T=1;
```

Scaling factors

```
lambdaF=logspace(log10(0.001),log10(10),100);
```

Type of IDA analysis

```
IM_DM='Sa_disp';
```

Mass

```
m=1;
```

Yield displacement

```
uy = 0.082*9.81/(2*pi/T)^2;
```

Post yield stiffness factor

```
pysf=0.01;
```

Fraction of critical viscous damping

```
ksi=0.05;
```

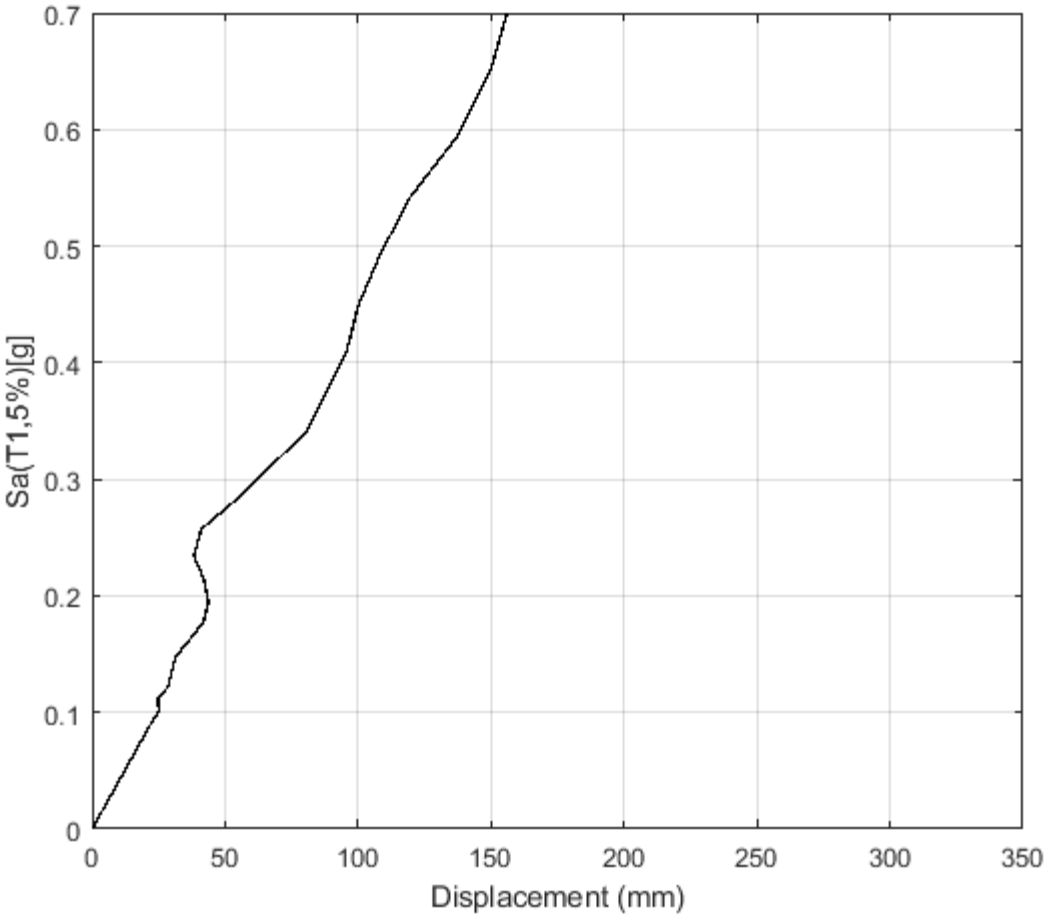
Apply OpenSeismoMatlab

```
S5=OpenSeismoMatlab(dt,xgtt,sw,T,lambdaF,IM_DM,m,uy,pysf,ksi);
```

Plot the displacement time histories

```
% Initialize figure
figure()
% Plot the response curve of the incremental dynamic analysis
plot(S5.DM*1000,S5.IM/9.81,'k','LineWidth',1)
% Finalize figure
grid on
xlabel('Displacement (mm)')
ylabel('Sa(T1,5%)[g]')
```

```
xlim([0,350])
ylim([0,0.7])
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the incremental dynamic analysis for multiple motions

Contents

Reference

Deng, P., Pei, S., van de Lindt, J. W., Liu, H., & Zhang, C. (2017). An approach to quantify the influence of ground motion uncertainty on elastoplastic system acceleration in incremental dynamic analysis. Advances in Structural Engineering, 20(11), 1744-1756.

Description

Figure 4(a) of the above reference contains the IDA curves of an elastoplastic SDOF system under the Consortium of Universities for Research in Earthquake Engineering (CUREE) GM suite (Krawinkler et al., 2001), which were constructed using the maximum acceleration. In this example, an arbitrary suite of strong ground motions is selected and the maximum acceleration IDA curves are constructed similar to Figure 4(a) of the above reference. The IDA curves of this example strongly resemble those of that figure.

Earthquake motions

Load data from a suite of earthquakes

```
GM={'Cape Mendocino.dat';
    'ChiChi.dat';
    'Christchurch2011HVPS_UP.dat';
    'Imperial Valley.dat';
    'Imperial_Valley_El_Centro_9_EW.dat';
    'Kobe.dat';
    'Kocaeli.dat';
    'San Fernando.dat';
    'Spitak.dat'};
n=size(GM,1);
dt=cell(n,1);
xgtt=cell(n,1);
for i=1:n
    fid=fopen(GM{i},'r');
    text=textscan(fid,'%f %f');
    fclose(fid);
    t=text{1,1};
    dt{i}=t(2)-t(1);
    xgtt{i}=text{1,2};
end
```

Setup parameters for IDA analysis

Switch

```
sw='ida';
```

Eigenperiod

```
T=1;
```

Scaling factors

```
lambdaF=logspace(log10(0.001),log10(10),100);
```

Type of IDA analysis

```
IM_DM='pgv_acc';
```

Mass

```
m=1;
```

Yield displacement

```
uy = 0.18*9.81/(2*pi/T)^2;
```

Post yield stiffness factor

```
pysf=0.01;
```

Fraction of critical viscous damping

```
ksi=0.05;
```

Algorithm to be used for the time integration

```
AlgID='U0-V0-Opt';
```

Set initial displacement

```
u0=0;
```

Set initial velocity

```
ut0=0;
```

Minimum absolute value of the eigenvalues of the amplification matrix

```
rinf=1;
```

Maximum tolerance for convergence

```
maxtol=0.01;
```

Maximum number of iterations per increment

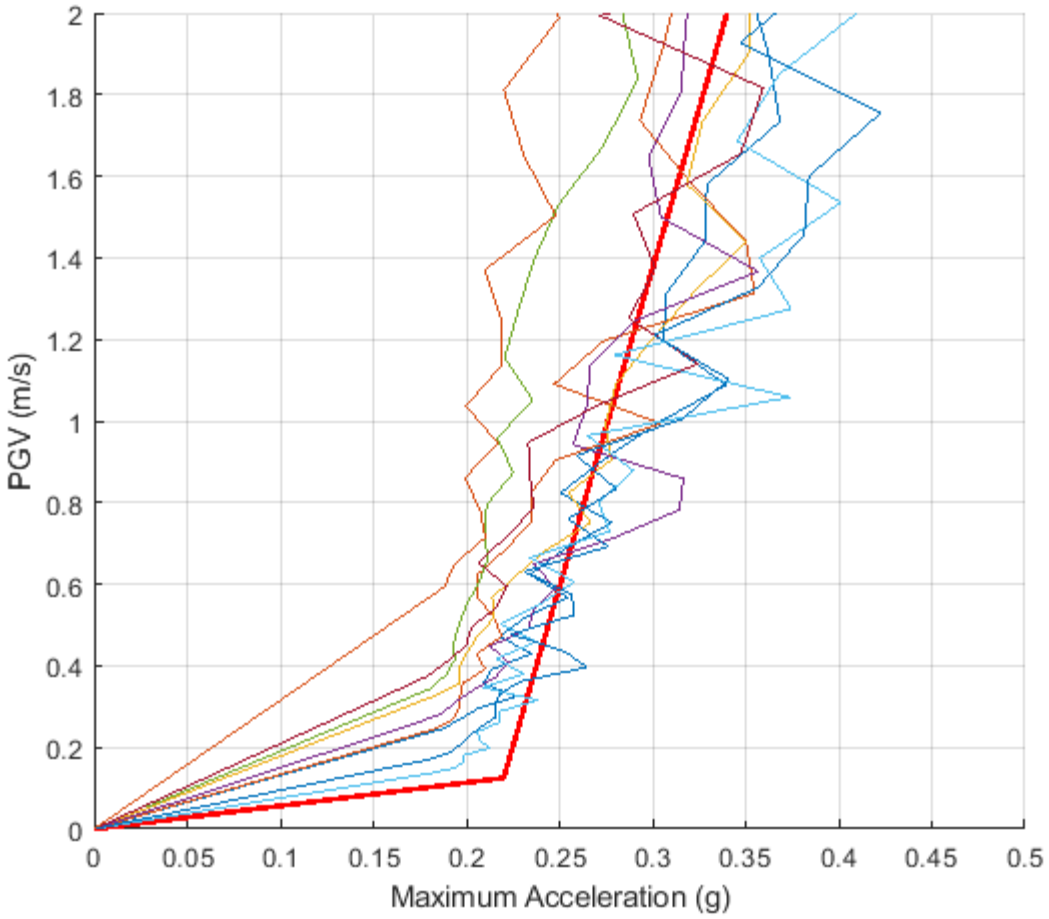
```
jmax=200;
```

Infinitesimal variation of acceleration

```
dak=eps;
```

Construct and plot the IDA curves in a loop

```
% Initialize figure
figure()
hold on
% Plot the red bold curve of Figure 4(a) of the above reference
plot([0,0.22,0.34],[0,0.125,2],'r','LineWidth',2)
for i=1:n
    % Apply OpenSeismoMatlab to calculate the ith IDA curve
    S1=OpenSeismoMatlab(dt{i},xgtt{i},sw,T,lambdaF,IM_DM,m,uy,pysf,ksi,AlgID,...
        u0,ut0,rinf,maxtol,jmax,dak);
    % Plot the ith IDA curve
    plot(S1.DM/9.81,S1.IM)
end
% Finalize figure
grid on
xlabel('Maximum Acceleration (g)')
ylabel('PGV (m/s)')
xlim([0,0.5])
ylim([0,2])
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the incremental dynamic analysis for ductility response

Contents

Reference

Vamvatsikos, D., & Cornell, C. A. (2002). Incremental dynamic analysis. Earthquake engineering & structural dynamics, 31(3), 491-514.

Description

The ductility response IDA curve of an elastoplastic SDOF system excited by the Loma Prieta, 1989, Halls Valley earthquake (component 090) is constructed at multiple levels of shaking, and compared to the curve shown in Figure 4(a) of the above reference. The SDOF system has T=1 sec and critical damping ratio 5%

Earthquake motion

Load earthquake data

```
GM='LomaPrietaHallsValley90.dat';
fid=fopen(GM,'r');
text=textscan(fid,'%f %f');
fclose(fid);
t=text{1,1};
dt=t(2)-t(1);
xgtt=text{1,2};
```

Perform IDA analysis

Switch

```
sw='ida';
```

Eigenperiod

```
T=1;
```

Scaling factors

```
lambdaF=logspace(log10(0.01),log10(30),100);
```

Type of IDA analysis

```
IM_DM='Sa_mu';
```

Mass

```
m=1;
```

Yield displacement

```
uy=0.25;
```

Post yield stiffness factor

```
pysf=0.01;
```

Fraction of critical viscous damping

```
ksi=0.05;
```

Algorithm to be used for the time integration

```
AlgID='U0-V0-Opt';
```

Set initial displacement

```
u0=0;
```

Set initial velocity

```
ut0=0;
```

Minimum absolute value of the eigenvalues of the amplification matrix

```
rinf=1;
```

Maximum tolerance for convergence

```
maxtol=0.01;
```

Maximum number of iterations per increment

```
jmax=200;
```

Infinitesimal variation of acceleration

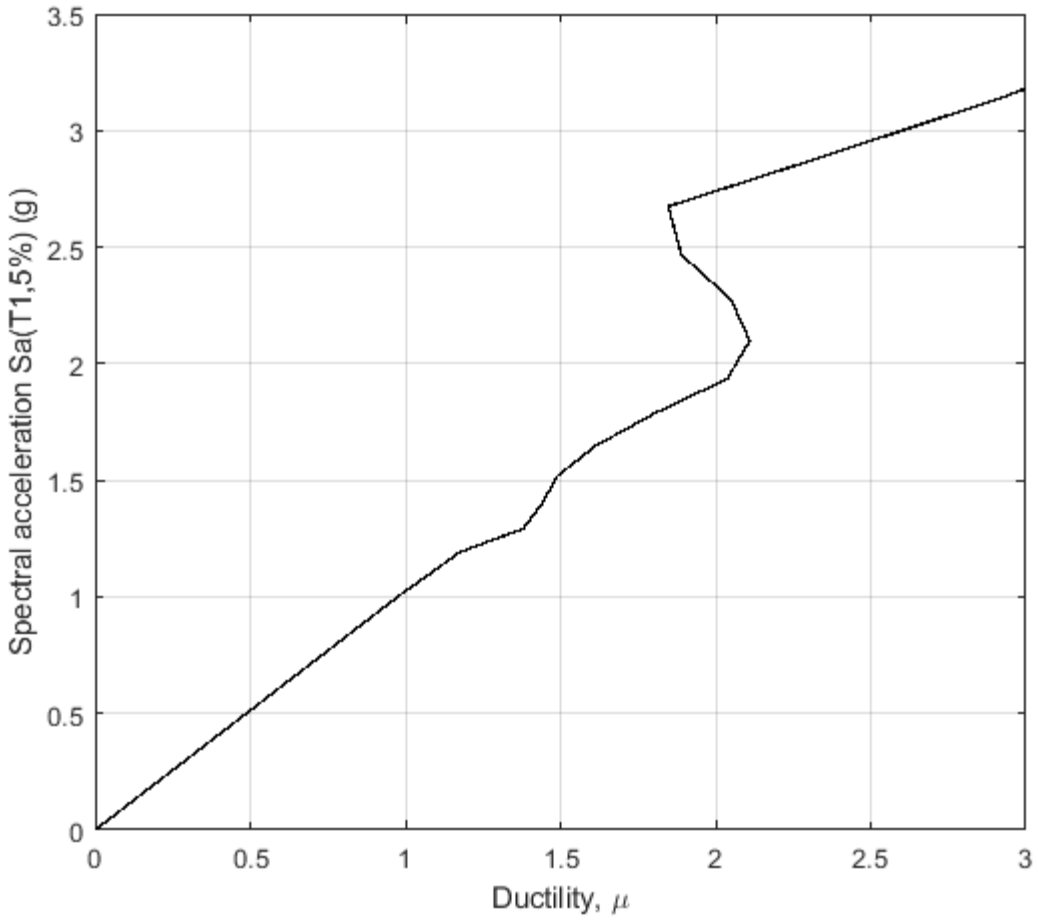
```
dak=eps;
```

Apply OpenSeismoMatlab

```
S1=OpenSeismoMatlab(dt,xgtt,sw,T,lambdaF,IM_DM,m,uy,pysf,ksi,AlgID,...
    u0,ut0,rinf,maxtol,jmax,dak);
```

Plot the IDA curve

```
% Initialize figure
figure()
% Plot the IDA curve
plot(S1.DM,S1.IM/9.81,'k','LineWidth',1)
% Finalize figure
grid on
xlabel('Ductility, \mu')
ylabel('Spectral acceleration Sa(T1,5%) (g)')
xlim([0,3])
ylim([0,3.5])
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the incremental dynamic analysis for ductility response

Contents

Reference

De Luca, F., Vamvatsikos, D., & Iervolino, I. (2011, May). Near-optimal bilinear fit of capacity curves for equivalent SDOF analysis. In Proceedings of the COMPDYN2011 Conference on Computational Methods in Structural Dynamics and Earthquake Engineering, Corfu, Greece.

Description

Figure 1(b) of the above reference presents the median IDA curves of SDOF systems with T=0.5sec. The actual capacity curve of the SDOF oscillator shown in FIgure 1(a) of the same reference (green line), has been fitted with an elastoplastic bilinear fit according to FEMA-440 (blue line). This fitting introduces an error (bias) which appears as the blue area in Figure 1(b), which is generally conservative. In this example two arbitrary acceleration time histories are selected, then the corresponding displacement response IDA curves are plotted, based on a SDOF system with suitably selected properties, based on Figure 1(a). It is shown that both curves approximately fall into the bias (blue area) of Figure 1(b) of the above reference.

Earthquake motions

Load data from two earthquakes

```
GM={'Imperial Valley'; % Imperial valley 1979
    'Cape Mendocino'};
n=size(GM,1);
dt=cell(n,1);
xgtt=cell(n,1);
for i=1:n
    fid=fopen([GM{i},'.dat'],'r');
    text=textscan(fid,'%f %f');
    fclose(fid);
    t=text{1,1};
    dt{i}=t(2)-t(1);
    xgtt{i}=text{1,2};
end
```

Setup parameters for IDA analysis

Switch

```
sw='ida';
```

Eigenperiod

```
T=0.5;
```

Scaling factors

```
lambdaF=logspace(log10(0.01),log10(30),100);
```

Type of IDA analysis

```
IM_DM='Sa_disp';
```

Yield displacement

```
uy=0.042;
```

Initial stiffness

```
k_hi=1000/uy;
```

Mass

```
m=k_hi/(2*pi/T)^2;
```

Post yield stiffness factor

```
pysf=0.01;
```

Fraction of critical viscous damping

```
ksi=0.05;
```

Algorithm to be used for the time integration

```
AlgID='U0-V0-Opt';
```

Set initial displacement

```
u0=0;
```

Set initial velocity

```
ut0=0;
```

Minimum absolute value of the eigenvalues of the amplification matrix

```
rinf=1;
```

Maximum tolerance for convergence

```
maxtol=0.01;
```

Maximum number of iterations per increment

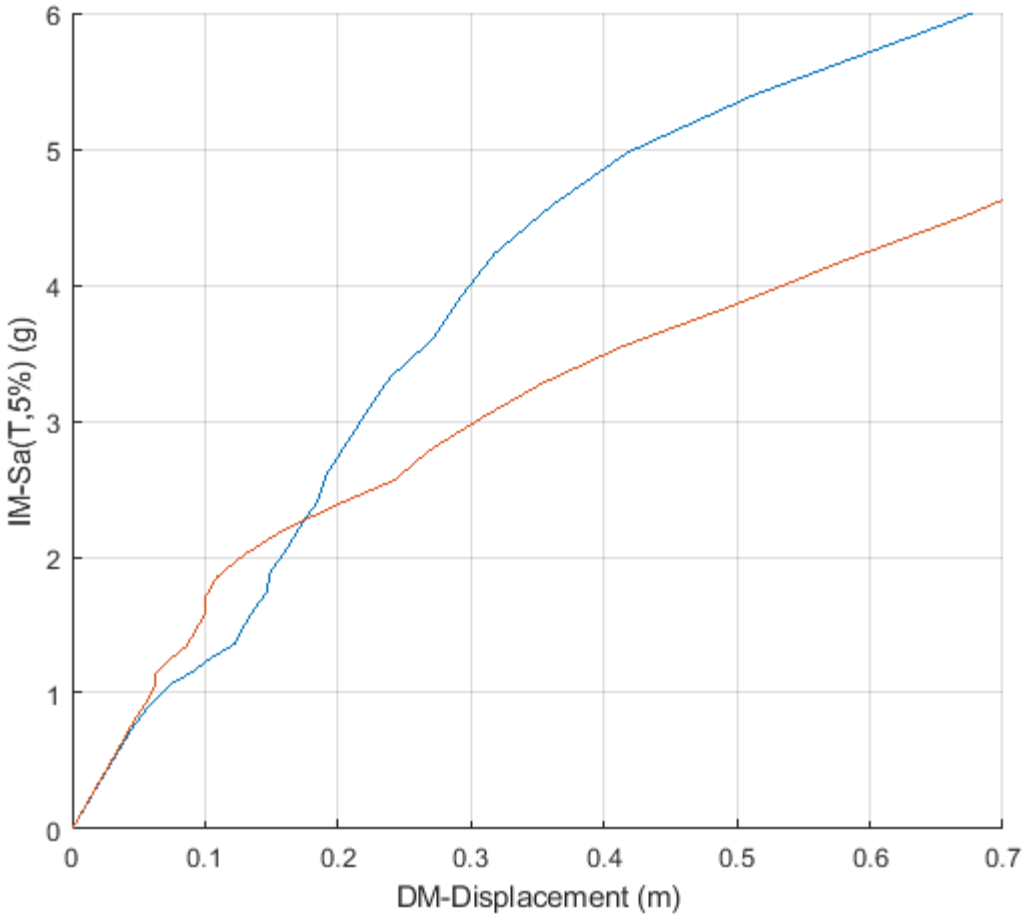
```
jmax=200;
```

Infinitesimal variation of acceleration

```
dak=eps;
```

Construct and plot the IDA curves in a loop

```
% Initialize figure
figure()
hold on
% Plot the IDA curves of Figure 1(b) of the above reference
for i=1:n
    S1=OpenSeismoMatlab(dt{i},xgtt{i},sw,T,lambdaF,IM_DM,m,uy,pysf,ksi,AlgID,...
        u0,ut0,rinf,maxtol,jmax,dak);
    plot(S1.DM,S1.IM/9.81)
end
% Finalize figure
grid on
xlabel('DM-Displacement (m)')
ylabel('IM-Sa(T,5%) (g)')
xlim([0,0.7])
ylim([0,6])
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the linear dynamic response of SDOF oscillator

Calculate the dynamic response of a linear SDOF oscillator. This example verifies Figure 6.6.1 in Chopra for Tn=1sec

Contents

Reference

Chopra, A. K. (2020). Dynamics of structures, Theory and Applications to Earthquake Engineering, 5th edition. Prenctice Hall.

Earthquake motion

Load earthquake data

```
dt=0.02;
fid=fopen('elcentro.dat','r');
text=textscan(fid,'%f %f');
fclose(fid);
xgtt=text{1,2};
```

Setup parameters for LIDA function

Eigenperiod

```
Tn=1;
```

Critical damping ratio

```
ksi=0.02;
```

Initial displacement

```
u0=0;
```

Initial velocity

```
ut0=0;
```

Algorithm to be used for the time integration

```
AlgID='U0-V0-Opt';
```

Minimum absolute value of the eigenvalues of the amplification matrix

```
rinf=1;
```

Calculate dynamic response

Calculate circular eigenfrequency

```
omega=2*pi/Tn;
```

Apply LIDA

```
[u,ut,utt] = LIDA(dt,xgtt,omega,ksi,u0,ut0,AlgID,rinf);
```

Results

Maximum displacement in cm

```
D=max(abs(u))*100
```

```
D =

   15.063275193038429
```

Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b

Verify the energy time history of SDOF oscillator

Calculate the time history of the strain energy and the energy dissipated by viscous damping and yielding of a linear and a nonlinear SDOF oscillator

Contents

Reference

Chopra, A. K. (2020). Dynamics of structures, Theory and Applications to Earthquake Engineering, 5th edition. Prenctice Hall.

Description

Figure 7.9.1 of the above reference is reproduced in this example, for both the linear elastic and the elastoplastic SDOF systems. The linear system has Tn=0.5 sec and ksi=5%, whereas the elastoplastic system has Tn=0.5 sec, ksi=5% and fybar=0.25.

Load earthquake data

Earthquake acceleration time history of the El Centro earthquake will be used (El Centro, 1940, El Centro Terminal Substation Building)

```
fid=fopen('elcentro.dat','r');
text=textscan(fid,'%f %f');
fclose(fid);
t=text{1,1};
dt=t(2)-t(1);
xgtt=text{1,2};
```

Setup parameters for NLIDABLKIN function for linear SDOF

Mass

```
m=1;
```

Eigenperiod

```
Tn=0.5;
```

Calculate the small-strain stiffness matrix

```
omega=2*pi/Tn;
k_hi=m*omega^2;
```

Assign linear elastic properties

```
k_lo=k_hi;
uy1=1e10;
```

Critical damping ratio

```
ksi=0.05;
```

Initial displacement

```
u0=0;
```

Initial velocity

```
ut0=0;
```

Algorithm to be used for the time integration

```
AlgID='U0-V0-Opt';
```

Minimum absolute value of the eigenvalues of the amplification matrix

```
rinf=1;
```

Maximum tolerance of convergence for time integration algorithm

```
maxtol=0.01;
```

Maximum number of iterations per integration time step

```
jmax=200;
```

Infinitesimal acceleration

```
dak=eps;
```

Calculate dynamic response of the linear SDOF

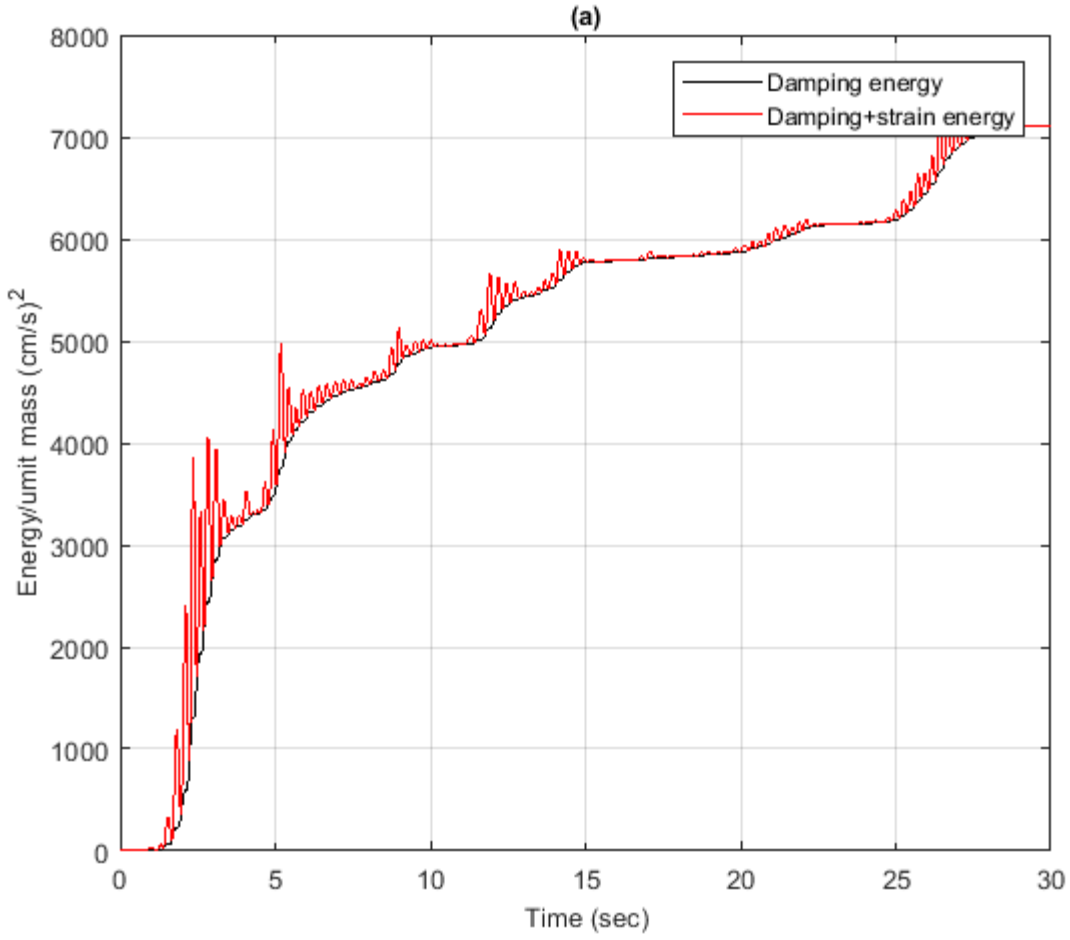
Apply NLIDABLKIN

```
[u,ut,utt,Fs,Ey,Es,Ed,jiter] = NLIDABLKIN(dt,xgtt,m,k_hi,k_lo,uy1,...
    ksi,AlgID,u0,ut0,rinf,maxtol,jmax,dak);
```

Plot the energy time history of the linear SDOF

Plot the damping energy and strain energy of the linearly elastic SDOF system. Convert from m to cm

```
figure()
plot(t',cumsum(Ed)*1e4,'k','LineWidth',1)
hold on
plot(t',cumsum(Ed)*1e4+Es*1e4,'r','LineWidth',1)
hold off
xlim([0,30])
ylim([0,8000])
xlabel('Time (sec)','FontSize',10);
ylabel('Energy/umit mass (cm/s)^2','FontSize',10);
title('(a)','FontSize',10)
grid on
legend('Damping energy','Damping+strain energy')
```



Setup parameters for NLIDABLKIN function for nonlinear SDOF

The properties of the nonlinear SDOF system are identical to those of the linear SDOF system, except for the yield displacement and the

post-yield stiffness.

Post yield stiffness

```
k_lo=0.01*k_hi;
```

normalized yield strength

```
fybar=0.25;
```

Yield displacement for nonlinear response

```
uy2=fybar*max(abs(u));
```

Calculate dynamic response of the nonlinear SDOF

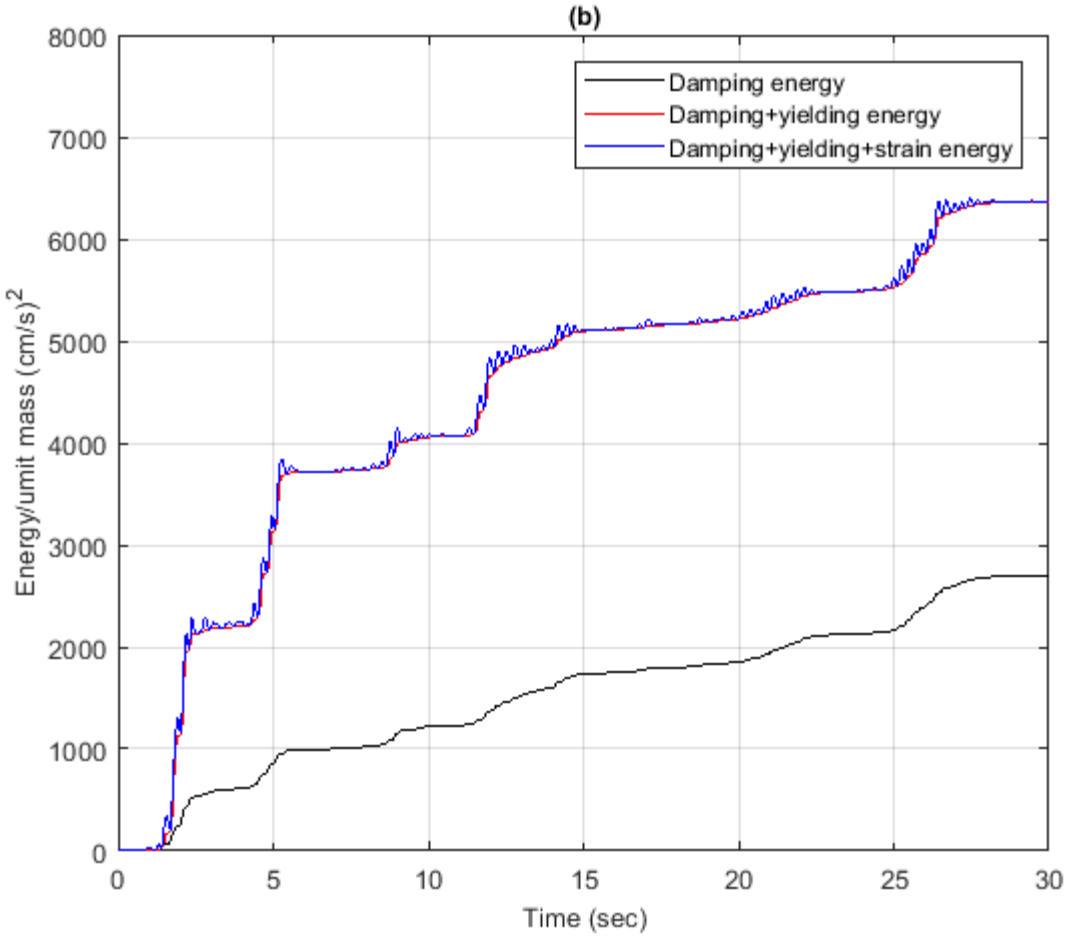
Apply NLIDABLKIN

```
[u,ut,utt,Fs,Ey,Es,Ed,jiter] = NLIDABLKIN(dt,xgtt,m,k_hi,k_lo,uy2,...
    ksi,AlgID,u0,ut0,rinf,maxtol,jmax,dak);
```

Plot the energy time history of the nonlinear SDOF

Plot the damping energy, the hysteretic energy and strain energy of the nonlinear SDOF system. Convert from m to cm.

```
figure();
plot(t',cumsum(Ed)*1e4,'k','LineWidth',1)
hold on
plot(t',cumsum(Ed)*1e4+cumsum(Ey)*1e4-Es*1e4,'r','LineWidth',1)
plot(t',cumsum(Ed)*1e4+cumsum(Ey)*1e4,'b','LineWidth',1)
hold off
xlim([0,30])
ylim([0,8000])
xlabel('Time (sec)','FontSize',10);
ylabel('Energy/umit mass (cm/s)^2','FontSize',10);
title('(b)','FontSize',10)
grid on
legend('Damping energy','Damping+yielding energy',...
    'Damping+yielding+strain energy')
```



Copyright

Copyright (c) 2018-2022 by George Papazafeiropoulos

- Major, Infrastructure Engineer, Hellenic Air Force
- Civil Engineer, M.Sc., Ph.D.
- Email: gpapazafeiropoulos@yahoo.gr

Published with MATLAB® R2017b