Table of Contents


```
function v = my_step(u,tf,x,v,dz,flag)

% wavefield extrapolation by phase shift in 3D
%
% use:
%   v = my_step(u,t,x,v,dz,dir)
%
% input:
%   u    -
%       flag = 1, data is given in frequency domain(default, if not specified
%                    by user)
%                - u(f,r,s)
%                is the 3D data volumn of a 2D seismic survy observed at the
%                surface z = 0 in frequency domain, first dimension is frequency,
%                second is receiver, third is source
%                - tf is the frequency coordinate
%
%       flag = 2, data is given in time domain
%                - u(t,r,s)
%                is the 3D data volumn of a 2D seismic survy observed at the
%                surface z = 0 in time domain, the first dimension is time, the
%                second is receiver, the third is source
%                - tf is the time coordinate
%   t    - time coordinates in seconds as column vector
%   x    - space coordinates in meters as row vector(as source and receiver
%          are at the same grid)
%   v    - velocity in m/s (scalar)
%   dz   - depth step in meters
%   flag-
%
% output:
%   v    - extrapolated wavefield as a vector
%
```

if length of t and x are not power of 2

pad zeros as fktran

transform to desired domain

```
if not(exist('flag','var'))
```

```
    flag = 1;
end
Ft = opDFT(size(u,1));
Fr = opDFT(size(u,2));
Fs = opDFT(size(u,3));
if flag == 1 %data is given in frequency domain
    It = opDirac(length(tf));
    F = opKron(Fs,Fr,It);
    U = F*vec(u);
    f = tf;
else % data is given in time domain
    F = opKron(Fs,Fr,Ft);
    U = F*vec(u);
    t = tf;
    fnyq = 1. / (2*(t(2)-t(1)));
    %nf = size(u,1);
    %f = linspace(0.,fnyq,nf)';
    f = [0:df:fnyq-df -fnyq:df:-df];
end

% w = f;
% compute kr(wavenumber for receiver)
dx = (x(2)-x(1));
fnyq = 1. / (2*(dx));
nf = size(u,2);
dff = 2*fnyq/nf;
fr = [0:dff:fnyq-dff -fnyq:dff:-dff];
kr = fr;
ks = kr;
```

```
Error using my_step (line 41)
Not enough input arguments.
```

compute DSR

```
[ff,kkr,kks]   = ndgrid(f,kr,ks);
DSR1        = 2*pi*sqrt((ff/v).^2-kkr.^2);
DSR2        = 2*pi*sqrt((ff/v).^2-kks.^2);
DSR1        = real(DSR1)+1i*abs(imag(DSR1));
DSR2        = -real(DSR2)+1i*abs(imag(DSR2));
DSR        = (DSR1 + DSR2);
%DSR        = -real(DSR)+1i*abs(imag(DSR));
```

propagate

```
U      = vec(exp(1i*abs(dz)*DSR)).*U;
```

transform back to time space space domain

```
F = opKron(Fs,Fr,Ft);
v      =   F'*vec(U);
```

if padding zeros

```
v = v(1:length(t),1:length(x));
```

Published with MATLAB® 7.13