Contents


```
function [rootx,rooty,rootz,it,success]=newton3D_exact(f,gradf,g,gradg,h,gradh,x0,y0,z0,maxit,tol,verbose)
```

```
% root=newton_exact(f,fprime)
%
% finds a set of roots corresponding to the function f,g (input as a handle)
% given a function which computes the derivative
%
% requires:  Gauss_elim.m and backsub.m
```

Need access to linear algebra routines for solves at each iteration

addpath ../linear_algebra;

Error checking of input

```
narginchk(6,12);   %check for correct number of inputs to function
if (nargin<7)
    maxit=100;       %maximum number of iterations allowed
end %if
if (nargin<8)
    tol=1e-6;        %how close to zero we need to get to cease iterations
end %if
if (nargin<9)
    verbose=false;
end %if
```

```
Error using newton3D_exact (line 16)
Not enough input arguments.
```

Make sure we don't start at an inflection point with zero derivative

```
[gradfx,gradfy,gradfz] = gradf(x0,y0,z0);
[gradgx,gradgy,gradgz] = gradg(x0,y0,z0);
[gradhx,gradhy,gradhz] = gradh(x0,y0,z0);
if (abs(min([gradfx,gradfy,gradfz,gradgx,gradgy,gradgz,gradhx,gradhy,gradhz]))<tol)    %this needs to really check inflection vs. saddle point; will fix later
    warning(' Attempting to start Newton iterations near an inflection/saddle point, you may wish to restart with a different guess...');
    x0 = x0 + 1;   %bump the guess a ways off of initial value to see if we can get anything sensible
    y0 = y0 + 1;
    z0 = z0 + 1;
end %if
```

Newton iterations

```
it = 1;
rootx = x0;
rooty = y0;
rootz = z0;
fval = f(rootx,rooty,rootz);
gval = g(rootx,rooty,rootz);
hval = h(rootx,rooty,rootz);
converged=false;
while(~converged && it<=maxit)
    [gradfx,gradfy,gradfz] = gradf(rootx,rooty,rootz);
    [gradgx,gradgy,gradgz] = gradg(rootx,rooty,rootz);
    [gradhx,gradhy,gradhz] = gradh(rootx,rooty,rootz);
    A=[gradfx,gradfy,gradfz; ...
        gradgx,gradgy,gradgz; ...
        gradhx,gradhy,gradhz];

    fvec=[fval;gval;hval];
    [Amod,ord]=Gauss_elim(A,-1*fvec);
    dxvec=backsub(Amod(ord,:));
    detA=prod(diag(Amod(ord,:)));
    if (abs(detA) < 1e-6)
        error(' Ended up at a point where Newton iteration is singular, try a different starting point')
    end %if

    rootx=rootx + dxvec(1);
    rooty=rooty + dxvec(2);
    rootz=rootz + dxvec(3);
    fval = f(rootx,rooty,rootz);                  % see how far off we are from zero...
    gval = g(rootx,rooty,rootz);
    hval = h(rootx,rooty,rootz);
    if (verbose)
```

```
        fprintf(' iteration: %d; x:  %f + %f i; y:  %f + %f i; f: %f, g:  %f\n',it,real(rootx),imag(rootx),real(rooty),imag(rooty),real(rootz),imag(rootz),fval,gval
    end %if
    it=it+1;
    converged=abs(fval)+abs(gval)+abs(hval)<tol;
end %while
it=it-1;

if (~converged)
    warning('Used max number of iterations, or derivative near zero...')
    success=false;
else
    success=true;
end %if
```

```
end %function
```

Published with MATLAB® R2020b