PFixNL.m

MATLAB, The Mathworks, Inc.

November 25, 2014

```
01          if (nonLine == 1)
02              U_i = U;
03              epsilon = 10^-6;
04              err = 1;
05              methode = 1;        % Modification de K a chaque pas de temps
06              for cacher = 1
07                  while err > epsilon
08                      % k = kres( U_i );
09                          K = K0;       % on retire l'ancienne contribution
10                          for i = 1:size(nonLinearite,2)
11                              k = nonLinearite(i).fonction(nonLinearite(i).dependanceEnU,U_i);
12                              if (isnan(k))
13                                  xfgchjkl
14                              end
15                              Kres = k*nonLinearite(i).matriceKUnit;
16                              K = K + Kres;
17                          end
18                          S = M + ( 1+alpha )*(C*gamma*dt + K*beta*dt^2);
19                      % S(Ui) * Ai = g1( U,Ui,V,F(n+1) );
20                          if (size(D,1))
21                               S = M + ( 1+alpha )*(C*gamma*dt + K*beta*dt^2);
22                               SD  = [S D';D zeros(size(D,1))];
23                               if (t>0 && methode == 1)
24                                   F = F - alpha*(K-Kn)*U; % en HHT on a besoin de l etape preceden
25                               end
26                               Fs  = F - C*Vp - K*Up ;
27                               Fsb = [Fs ; conditionA(:,t+1)];
28                               AL  = SD\Fsb;
29                               Ai = AL(1:size(A,1),1);
30                               Landa = AL((size(A,1)+1):end,1);
31                          else
32                              Ai = S\( F - C*Vp - K*Up );
33                          end
34                      % Ui = U + g3( V,A,Ai )
35                          U_ip = U_i;     % terme precedent
36                          U_i = U +  dt*V + (dt^2)*( (1/2 - beta)*A  + beta*Ai );
37                      % convergence
38                          if (U_ip==U_i)
39                              err = 0;
40                          else
41                              err = norm(U_i-U_ip)/ ( norm(U_i)+norm(U_ip) );
42                          end
43                  end
44                  U_i
45              end
```

```
46
47              if (t>0)
48                   U = U_i;
49                   V = V + dt*(1-gamma)*A + dt*gamma*Ai;
50                   A = Ai;
51
52                   if (verif)     % correction de l erreur d integration, impossible si les deplace
53                       [i,~]=find(D');
54                       V(i,1) = conditionV(:,t+1);
55                       U(i,1) = conditionU(:,t+1);
56                   end
57              end
58
59          else
```