Finite Element Implementation of Quasi-static Phase Field Brittle Fracture

Wenqiang Fang

March 16, 2020

1 Energy Functional

The regularized energy functional given in Kaushik's Manuscript reads,

$$\boldsymbol{E}_l(\boldsymbol{u},\phi) = \int_{\mathcal{B}} \left[(1-\phi)^2 + \mathcal{K}\right] \Psi_0(\boldsymbol{u})\, d\Omega - \int_{\mathcal{B}_t} \bar{\boldsymbol{t}}\cdot\boldsymbol{u}\, dS + \int_{\mathcal{B}} \frac{g_c}{2}\left(\frac{\phi^2}{l} + l||\nabla\phi||^2\right) d\Omega, \tag{1}$$

where $\phi(\boldsymbol{X})$ is the damage field ($\phi = 0$ means intact while $\phi = 1$ is fully damaged), $\Psi_0(\boldsymbol{u})$ is the strain energy density. For linear elastic solid, we have

$$\Psi_0(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{\sigma}(\epsilon) : \boldsymbol{\epsilon} = \frac{1}{2}\boldsymbol{\epsilon} : \mathbb{C} : \boldsymbol{\epsilon}, \tag{2}$$

where $\boldsymbol{\epsilon} = \frac{1}{2}(\boldsymbol{u} + \boldsymbol{u}^T)$ is the infinitesimal strain tensor.

Substituting (2) into functional (1) and taking variation of the resulting equation, we obtain

$$\begin{aligned}\boldsymbol{\delta E}_l(\boldsymbol{u},\phi,\boldsymbol{\delta u},\delta\phi) = &\int_{\mathcal{B}} \left[(1-\phi)^2 + \mathcal{K}\right] \boldsymbol{\epsilon} : \mathbb{C} : \boldsymbol{\delta\epsilon}\, d\Omega - \int_{\mathcal{B}_t} \bar{\boldsymbol{t}}\cdot\boldsymbol{\delta u}\, dS \\ &+ \int_{\mathcal{B}} -2(1-\phi)\left(\frac{1}{2}\boldsymbol{\epsilon} : \mathbb{C} : \boldsymbol{\epsilon}\right)\delta\phi\, d\Omega + \int_{\mathcal{B}} g_c\left(\frac{\phi}{l}\delta\phi + l\nabla\phi\cdot\nabla\delta\phi\right) d\Omega.\end{aligned} \tag{3}$$

2 Finite element equations

We discretize the displacement field and damage field as

$$u_i(\boldsymbol{X}) = \Sigma_{a=1}^{n_{\text{np}}} N^a(\boldsymbol{X}) u_i^a, \tag{4}$$

$$\phi(\boldsymbol{X}) = \Sigma_{a=1}^{n_{\text{np}}} N^a(\boldsymbol{X}) \phi^a. \tag{5}$$

We calculate the gradient of u_i and ϕ as

$$u_{i,j}(\boldsymbol{X}) = \Sigma_{a=1}^{n_{\text{np}}} \frac{dN^a}{dX_j}(\boldsymbol{X}) u_i^a, \tag{6}$$

$$\phi_{,i}(\boldsymbol{X}) = \Sigma_{a=1}^{n_{\text{np}}} \frac{dN^a}{dX_i}(\boldsymbol{X}) \phi^a, \tag{7}$$

We interpolate the virtual displacement field and virtual damage field in exactly the same way. Substituting the interpolated fields into the virational form (3), we find that

$$\begin{aligned}\boldsymbol{\delta E}_l^h(u_i^a,\phi^a,\delta u_i^b,\delta\phi^b) &= \int_{\mathcal{B}}\left[(1-\phi)^2+\mathcal{K}\right]\mathbb{C}_{ijkl}\frac{dN^a}{dX_j}\frac{dN^b}{dX_l}u_i^a\delta u_k^b\,d\Omega - \int_{\mathcal{B}_t}\bar{t}_k\cdot N^b\delta u_k^b\,dS\\ &+\int_{\mathcal{B}}-2(1-\phi)\left(\frac{1}{2}\boldsymbol{\epsilon}:\mathbb{C}:\boldsymbol{\epsilon}\right)N^b\delta\phi^b\,d\Omega+\int_{\mathcal{B}}g_c\left(\frac{\phi}{l}N^b\delta\phi^b+l\phi^a\frac{dN^a}{dX_i}\frac{dN^b}{dX_i}\delta\phi^b\right)d\Omega\\ &=\int_{\mathcal{B}}\left[(1-\phi)^2+\mathcal{K}\right]\mathbb{C}_{ijkl}\frac{dN^a}{dX_j}\frac{dN^b}{dX_l}u_i^a\delta u_k^b\,d\Omega-\int_{\mathcal{B}_t}\bar{t}_k\cdot N^b\delta u_k^b\,dS\\ &+\int_{\mathcal{B}}\left[\left(\frac{g_c}{l}N^aN^b+g_cl\frac{dN^a}{dX_i}\frac{dN^b}{dX_i}+2\mathcal{H}N^aN^b\right)\phi^a-2\mathcal{H}N^b\right]\delta\phi^b\,d\Omega,\end{aligned}\tag{8}$$

where $\mathcal{H}=\Psi_0(\boldsymbol{u})=\frac{1}{2}\boldsymbol{\epsilon}:\mathbb{C}:\boldsymbol{\epsilon}$ is the local history field of stain energy density. It can be computed as

$$\mathcal{H}=\mathbb{C}_{ijkl}\frac{dN^a}{dX_j}\frac{dN^b}{dX_l}u_i^au_k^b.\tag{9}$$

The above variation should vanish for arbitrary nodal values for $\delta\boldsymbol{u}$ and $\delta\phi$. We split the displacement and damage field, using staggered scheme to solve the above equation. Therefore, we can re-write the virtual work equations in Matrix form as

$$(K^u_{aibk}u_i^a-F^u_{bk})\delta u_k^b=0,\tag{10}$$

and

$$(K^\phi_{ab}\phi^a-F^\phi_b)\delta\phi^b=0,\tag{11}$$

where

$$K^u_{aibk}=\int_{\mathcal{B}}\left[(1-\phi)^2+\mathcal{K}\right]\mathbb{C}_{ijkl}\frac{dN^a}{dX_j}\frac{dN^b}{dX_l}\,d\Omega,\tag{12}$$

$$F^u_{bk}=\int_{\mathcal{B}_t}\bar{t}_k\cdot N^b\,dS,\tag{13}$$

$$K^\phi_{ab}=\int_{\mathcal{B}}\left[\left(\frac{g_c}{l}+2\mathcal{H}\right)N^aN^b+g_cl\frac{dN^a}{dX_i}\frac{dN^b}{dX_i}\right]d\Omega,\tag{14}$$

$$F^\phi_b=\int_{\mathcal{B}}2\mathcal{H}N^b\,d\Omega.\tag{15}$$

From the Global Stiffness and Force matrix, it is straight forward to derive the expression for Elemental Stiffness and Force matrix. Here we neglect the details in the element-wise expression.

3 Newton Raphson scheme

In the above section, we show how to reduce equation (8) to a linear system. In this section, we are presenting a more general approach to solve the equation using Newton Raphson scheme. For a linear problem, Newton Raphson method takes only one iteration to converge, which makes it identical to LinearSolve (command in Mathematica). The advantage of applying Newton Raphson scheme is that it simplifies the implementation of Dirichlet boundary condition. Furthermore, if we want to add nonlinearity into our problem, it is straightforward. We just need to change the constitutive law.

We start with some initial guess for u_i^a and ϕ^a, which satisfy Dirichlet boundary conditions. Then we attempt the correct this guess by adding increments du_i^a and $d\phi^a$. We want the increments to satisfy

$$\int_{\mathcal{B}} \left[(1-\phi)^2 + \mathcal{K}\right] \mathbb{C}_{ijkl} \frac{dN^a}{dX_j} \frac{dN^b}{dX_l} (u_i^a + du_i^a)\, d\Omega - \int_{\mathcal{B}_t} \bar{t}_k \cdot N^b \, dS = 0, \tag{16}$$

$$\int_{\mathcal{B}} \left\{ \left[\left(\frac{g_c}{l} + 2\mathcal{H} \right) N^a N^b + g_c l \frac{dN^a}{dX_i} \frac{dN^b}{dX_i} \right] (\phi^a + d\phi^a) - 2\mathcal{H} N^b \right\} d\Omega = 0. \tag{17}$$

Noting that the we keep ϕ^a unchanged in (16) and u_i^a unchanged in (17) to decouple du_i^a and $d\phi^a$.

Since equation (16) and (17) are already linear on du_i^a and $d\phi^a$, respectively, we do not need to linearize them. In general, linearization are necessary.

We re-write the above equations in Matrix form as

$$K^u_{aibk} du_i^a + R^u_{bk} - F^u_{bk} = 0, \tag{18}$$

and

$$K^\phi_{ab} d\phi^a + R^\phi_b - F^\phi_b = 0, \tag{19}$$

where K^u_{aibk}, K^ϕ_{ab}, F^u_{bk}, F^ϕ_b are same as before, with

$$R^u_{bk} = \int_{\mathcal{B}} \left[(1-\phi)^2 + \mathcal{K}\right] \mathbb{C}_{ijkl} \frac{dN^a}{dX_j} \frac{dN^b}{dX_l} u_i^a \, d\Omega, \tag{20}$$

and

$$R^\phi_b = \int_{\mathcal{B}} \left[\left(\frac{g_c}{l} + 2\mathcal{H} \right) N^a N^b + g_c l \frac{dN^a}{dX_i} \frac{dN^b}{dX_i} \right] \phi^a \, d\Omega. \tag{21}$$

4 Viscosity

We add a viscous term $\frac{\eta}{2\tau}(\phi - \phi_n)^2$ into the functional (1) to stabilize the numerical treatment. The modified expression for stiffness, force and residual after the change are given by

$$K^\phi_{ab} = \int_{\mathcal{B}} \left[\left(\frac{g_c}{l} + 2\mathcal{H} + \frac{\eta}{\tau} \right) N^a N^b + g_c l \frac{dN^a}{dX_i} \frac{dN^b}{dX_i} \right] d\Omega, \tag{22}$$

$$F^\phi_b = \int_{\mathcal{B}} \left(2\mathcal{H} + \frac{\eta}{\tau} \phi_n^a N^a \right) N^b \, d\Omega, \tag{23}$$

$$R^\phi_b = \int_{\mathcal{B}} \left[\left(\frac{g_c}{l} + 2\mathcal{H} + \frac{\eta}{\tau} \right) N^a N^b + g_c l \frac{dN^a}{dX_i} \frac{dN^b}{dX_i} \right] \phi^a \, d\Omega.. \tag{24}$$

5 Implementation

Some important implementation details are list below:

1. ApplyBC($t0\_$) defines both the Dirichlet and Neumann boundary condition. The functions $\hat{u}, \hat{v}, \hat{w}$ are the prescribed Dirichlet boundary conditions and t_x, t_y, t_z are the applied Neumann boundary conditions.

2. SetUpDirichletBC and SetUpNeumannBC sets up the node numbers, degree of freedoms and corresponding prescribed values.

3. Edges stores all the edges (in 2D) where Neumann boundary conditions are prescribed.
4. UMat = ReplacePart[UMat, DirichletNode] actually prescribes the Dirichlet boundary conditions.
5. Map[ApplyTraction, Edges] actually prescribes the Neumann boundary conditions.
6. IDno stores all the non-Dirichlet equation numbers.
7. PlotMeshBC[ShowNumbers −> True,ShowBCs −> "DirichletBC"] plot the mesh with numbers on nodes and elements and Dirichlet boundary conditions marked in colors.
8. ComputeRF[] computes the reaction forces on the Neumann boundaries.

6 Summary on Mar 16

I am going to suspend for a while on this project. I have achieved the following goal so far:

1. Extend the 3D elasticity code to adapt 2D computation.
2. Change all component calculation to tensor calculation.
3. Include Neumann boundary condition (only works for 2D simple geometry)
4. Change the linear solver to Newton-Raphson nonlinear solver.
5. Mesh and boundary condition visualization.
6. Parallel computation on local machine.
7. Parallel computation on remote nodes (ccv).
8. Staggered scheme for Phase field fracture.
9. Add viscosity for stabilization.
10. Adaptive step loading.
11. Validation of the code for simple mode-I fracture.

The latest version of elastic code is given by Mar15_Elasticity.nb

The latest version of phase field code is given by Visco_Mar11

Miehe's model is in Mar9_KinkAngle.nb