A second-order cone interior-point method for initially rigid cohesive fracture*

Stephen A. Vavasis[a,*], Katerina D. Papoulia[b], Mohammadreza Hirmand[c]

[a] *Department of Combinatorics & Optimization, University of Waterloo, 200 University Ave. W., Waterloo, Ontario, N2L 3G1, Canada.*
[b] *Department of Mechanical & Mechatronics Engineering and Department of Applied Mathematics, University of Waterloo, 200 University Ave. W., Waterloo, Ontario, N2L 3G1, Canada*
[c] *Department of Mechanical & Mechatronics Engineering, University of Waterloo, 200 University Ave. W., Waterloo, Ontario, N2L 3G1, Canada*

Abstract

Initially rigid cohesive fracture is among the few techniques able to model complex and branching fracture with a sharp (nonsmeared) representation of the crack. Implicit time-stepping schemes are often favored in mechanics due to their ability to take larger time steps, but it is challenging to include an initially rigid cohesive model in an implicit scheme because the initiation of fracture corresponds to a nondifferentiability of the underlying potential. In this work, an interior-point method is proposed for implicit time stepping of initially rigid cohesive fracture. It uses techniques developed for convex second-order cone programming for the nonconvex problem at hand. The underlying cohesive model is taken from Papoulia (2017) and is based on a nondifferentiable energy function. That previous work proposed an algorithm based on successive smooth approximations to the nondifferential objective for solving the resulting optimization problem. It is argued herein that cone programming can capture the nondifferentiability without smoothing, and the resulting cone formulation is amenable to interior-point algorithms. A further benefit of the formulation is that other conic inequality constraints are straightforward to incorporate. Computational results are provided showing that certain contact constraints can be easily handled and that the method is practical.

1. A nondifferentiable energy model for cohesive fracture

In this section, we review prior work by Papoulia [15] as well as related works on nondifferentiable energy models for fracture. We assume throughout that temperature effects can be neglected so that consideration of thermodynamics reduces to mechanical potential and kinetic energy.

Let $\Omega \subset \boldsymbol{R}^{n_{\rm dim}}$ ($n_{\rm dim} = 2, 3$) denote the initial configuration of the body under consideration. Let $\mathcal{S} \subset \Omega$ be a union of $(n_{\rm dim} - 1)$-dimensional surfaces (when $n_{\rm dim} = 3$) or curves (when $n_{\rm dim} = 2$) that may each cut across the entire domain. Let $u : \Omega \to \boldsymbol{R}^{n_{\rm dim}}$ be the displacement field, assumed to be differentiable except possibly for jumps on $\mathcal{S}$.

In this model of solid mechanics and fracture, two potential energies exist, one associated with the bulk model and one with a network of surfaces $\mathcal{S}$ inside the domain that serve as potential sites of fracture. Thus, the mechanical potential has the form:

$$\mathcal{E} = \int_\Omega \Psi(u)\, dV + \int_{\mathcal{S}} \Phi([\![u]\!])\, dS + \int_\Omega f_1 \cdot u + \int_{\partial\Omega} f_2 \cdot u \tag{1}$$

where $[\![u]\!]$ denotes the jump in u across the surface. Here, Ψ corresponds to the strain-energy density function while Φ corresponds to energy density of new surfaces (fracture). The final two terms correspond

*Supported in part by an NSERC Discovery Grant.
*Corresponding author.

to body loads and traction loads respectively. Note that the use of a potential energy functional implies reversibility; we return to the matter of reversibility below. Later on, we will add another term to the optimization formulation to account for momentum. In addition we will impose displacement boundary conditions and inequality constraints. The latter will be used to model contact and a no-interpenetration requirement for the mesh.

We further stipulate that $\Phi([\![u]\!])$ is a nondifferentiable function of $[\![u]\!]$ when $[\![u]\!] = 0$. As explained in [15], this is a mandatory ingredient of the formulation. Because of the nondifferentiability at $[\![u]\!] = 0$, no jumps in u will occur across any surface until a positive finite level of loading occurs.

The above model falls into the category of "cohesive zone" models [1, 18, 4] because it accounts for crack propagation with explicit representation of a crack surfaces and an associated displacement-traction relation (which is obtained as a derivative of Φ for nonzero values of $[\![u]\!]$). Furthermore, it falls into the category of "initially rigid" cohesive zone models because of the property that there is no crack opening until a specific positive finite load level is attained. Initially rigid models are preferred over the alternative "initially elastic" models in problems where the crack path is not known a priori. Inclusion of a network of initially elastic surfaces would lower the global stiffness of Ω; as the number of crosscutting surfaces in $\mathcal{S}$ tends to infinity, the global stiffness is driven to 0. In contrast, there is no limit to how much surface area may be encompassed by $\mathcal{S}$ in (1) for the class of nondifferentiable potentials $\Phi([\![u]\!])$ proposed in [15].

In [16], it was argued that unless significant care is taken in designing the algorithm, methods for initially rigid cohesive fracture are likely to be "time discontinuous." The issue is that after space discretization, a system of ODE's for nodal values of the displacement u and other quantities arises. The forcing function of these ODE's may have a discontinuous dependence on u, and this leads to nonconvergent or unreliable numerical methods. In [15], the problems of time discontinuity are sidestepped because the modeling technique does not correspond to a system of ODE's at all—the usual step of passing to a weak form does not apply because the potential is nondifferentiable. Instead, the method corresponds to a sequence of physically based energy minimization operations.

The formulation (1) thus reduces the problem of modeling fracture to a sequence of optimization problems. These are infinite dimensional problems, but they are reduced to finite-dimensional optimization using finite element analysis as discussed in Section 3. This problem was solved in [15] using a continuation method. Hirmand and Papoulia [8] solve it using a Nitsche discontinuous Galerkin method (see also the related work by Radovitzky et al. [17]). in which the multipliers of the optimization problem are interpreted as stresses at the crack surface. The contribution of this paper is a solution method for the optimization problem (1) using a novel interior-point method. A key step in the development, as explained in Section 5, is to recast a certain equation that appears in the optimization problem an inequality. This technique is commonplace in the optimization literature but is new (as far as we know) to fracture mechanics. General background on interior-point methods is provided in Section 2. A computational experiment with our method is provided in Section 9.

We conclude this section with a discussion of related literature as well as an explanation of extensions of the method. Other than Papoulia [15] and Hirmand and Papoulia [8], the most closely related work is Lorentz's [11] method, which also treats initially rigid fracture using a potential like (1) for the same reasons as us. Lorentz does not use an optimization method per se but rather considers the subdifferential of (1) as a generalization of a system of equations for generating a time step. Slightly more distantly related to the present work is the phase-field method of modeling fracture [3]. In this case, energy minimization is also invoked, but the functional pertains to a smeared crack location rather than a sharp surface. As a consequence, a sharp representation of the crack must be determined *a posteriori*, although some authors e.g., Geelen et al. [7], Wang and Waisman [21], have shown recently that a sharp representation of the crack can be directly coupled to a phase-field model.

The method as described so far is reversible. As Papoulia [15] explains, irreversibility may be incorporated via the additional dependence of Φ in (1) on a damage variable. Such an approach is also incorporated in this work as detailed in Section 3.

2. Interior-point algorithms

In this section we present general background on interior-point methods. For more in-depth treatment, see, e.g., [23]. The application of these methods to initially rigid cohesive fracture is provided in Section 5.

A *closed convex cone* is defined to be a set $K \subset \boldsymbol{R}^n$ with the properties that (i) K is closed, (ii) K is convex, (iii) K is a cone, i.e., $\boldsymbol{x} \in K \Rightarrow \lambda \boldsymbol{x} \in K$ for all $\lambda \geq 0$, and (iv) $\boldsymbol{0} \in K$ (i.e., $K \neq \emptyset$).

Two important special examples of closed convex cones are $\boldsymbol{R}^+_n = \{\boldsymbol{x} \in \boldsymbol{R}^n : x_i \geq 0 \, \forall i = 1, \ldots, n\}$, the *nonnegative orthant*, and $C^n_2 = \{\boldsymbol{x} \in \boldsymbol{R}^n : x_1 \geq \|\boldsymbol{x}(2:n)\|\}$, the *second-order cone*.

The two cones mentioned in the previous paragraph both have standard *self-concordant barrier functions*. For $\boldsymbol{R}^+_n$, the standard self-concordant barrier function is $\phi_{\mathrm{NNO}}(\boldsymbol{x}) = -\sum_{i=1}^n \log(x_i)$. For C^n_2, the standard self-concordant barrier function is $\phi_{\mathrm{SOC}}(\boldsymbol{x}) = -\frac{1}{2}\log(x_1^2 - x_2^2 - \cdots - x_n^2)$. We regard these functions as taking on the value infinity outside the relevant cones. These functions have the property that they are strictly convex functions on the interior of their respective cones, and they tend to infinity as the boundary of the cone is approached. "Self concordance" involves two other technical properties; see [13].

If $K_1 \subset \boldsymbol{R}^{n_1}, \ldots, K_r \subset \boldsymbol{R}^{n_r}$ are all closed convex cones with barrier functions $\phi_1, \ldots, \phi_r$, then $K_1 \times \cdots \times K_r$ is also a closed convex cone, and its barrier function is $\phi_1(\boldsymbol{x}_1) + \cdots + \phi_r(\boldsymbol{x}_r)$ for $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r) \in K_1 \times \cdots \times K_r$.

Consider the optimization problem

$$\begin{array}{rl} \min_x & f(\boldsymbol{x}) \\ \text{s.t.} & \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}, \\ & \boldsymbol{x} \in K, \end{array}$$

where K is a closed convex cone that has a barrier function $\phi(\boldsymbol{x})$. This problem may be solved as follows. Let $\mu_1 \equiv \mu_{\mathrm{init}}, \mu_2, \ldots$ be a decreasing sequence of positive parameters tending to 0. Then for $k = 1, 2 \ldots$ we solve

$$\begin{array}{rl} \min_{\boldsymbol{x}} & f(\boldsymbol{x}) + \mu_k \phi(\boldsymbol{x}) \\ \text{s.t.} & \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}, \end{array} \tag{2}$$

an equality-constrained optimization problem, and we define $\boldsymbol{x}_k$ to be the optimizer or approximate optimizer. On iteration $k+1$, we use $\boldsymbol{x}_k$ as the initial guess for the optimization algorithm, which is commonly Newton's method. In other words, we iteratively solve a sequence of equality-constrained optimization problems. This method is called a *primal* or *primal-only* interior-point method.

The case most commonly studied in the literature is the case when $f(\boldsymbol{x})$ is the linear function $\boldsymbol{c}^T\boldsymbol{x}$ and convex and the equality constraints are linear: $\boldsymbol{g}(\boldsymbol{x}) \equiv A\boldsymbol{x} - \boldsymbol{b}$ for some matrix A and vector $\boldsymbol{b}$. In this case, there is an extensive theory guaranteeing convergence to a global optimizer for the above algorithm for a suitable sequence of weights $\mu_1, \mu_2, \ldots$. See, e.g., [13].

We can also define a *primal-dual* method as follows. We first write the first-order (Lagrange or KKT) optimality condition for (2), which is,

$$\nabla f(\boldsymbol{x}^*) + \mu_k \nabla \phi(\boldsymbol{x}^*) + J(\boldsymbol{x}^*)^T \boldsymbol{\lambda} = \boldsymbol{0}, \tag{3}$$

where $\boldsymbol{x}^*$ is the optimizer of (2), $J(\boldsymbol{x}^*)$ denotes the first derivative (Jacobian matrix) of $\boldsymbol{g}(\boldsymbol{x})$, and $\boldsymbol{\lambda}$ is the Lagrange multiplier.

Assume k in the previous item is fixed for now. In the case of $K = \boldsymbol{R}^n_+$ so that $\phi_{\mathrm{NNO}}(\boldsymbol{x}) = -\sum_{i=1}^n \log(x_i)$ and

$$\nabla \phi(\boldsymbol{x}) = \begin{pmatrix} -1/x_1 \\ \vdots \\ -1/x_n \end{pmatrix},$$

we define dual variable $s_i = \mu_k / x_i$ for $i = 1, \ldots, n$. Then the optimality condition (3), combined with feasibility and with a rearrangement of the definition of s_i yields the following system of equations:

$$\begin{array}{rcll} \nabla f(\boldsymbol{x}) - \boldsymbol{s} + J(\boldsymbol{x})^T \boldsymbol{\lambda} & = & \boldsymbol{0}, & \\ \boldsymbol{g}(\boldsymbol{x}) & = & \boldsymbol{0}, & \\ x_i s_i & = & \mu_k, & \forall i = 1, \ldots, n. \end{array}$$

The final group of equations is called "complementarity". It may be rewritten $\boldsymbol{x} \circ \boldsymbol{s} = \mu_k \boldsymbol{e}$, where "$\circ$" is called the *Jordan product* for $\boldsymbol{R}^n_+$, and $\boldsymbol{e}$ denotes the vector of all 1's. The Jordan product is defined exactly as: the ith entry of $\boldsymbol{x} \circ \boldsymbol{s}$ is $x_i s_i$. This notation also implies that $\boldsymbol{x}, \boldsymbol{s}$ are in the cone, i.e., $x_i \geq 0$ and $s_i \geq 0$ for all $i = 1, \ldots, n$.

In the case of C^n_2, the gradient of the barrier function is

$$\nabla \phi_{\mathrm{SOC}}(\boldsymbol{x}) = \begin{pmatrix} -x_1/d \\ x_2/d \\ \vdots \\ x_n/d \end{pmatrix},$$

where $d = x_1^2 - x_2^2 - \cdots - x_n^2$. Then we define dual variables $s_1 = \mu_k x_1/d$, $s_2 = -\mu_k x_2/d$, $\ldots$, $s_n = -\mu_k x_n/d$. Note that, assuming $\boldsymbol{x} \in C^n_2$, it also follows from these formulas that $\boldsymbol{s} \in C^n_2$. In this case, the optimality condition (3) plus feasibility and the definition of $\boldsymbol{s}$ can be written:

$$\begin{aligned} \nabla f(\boldsymbol{x}) - \boldsymbol{s} + J(\boldsymbol{x})^T \boldsymbol{\lambda} &= \boldsymbol{0}, \\ \boldsymbol{g}(\boldsymbol{x}) &= \boldsymbol{0}, \\ \boldsymbol{x} \circ \boldsymbol{s} &= \mu_k \boldsymbol{e}. \end{aligned}$$

Here, for C^n_2, the Jordan product $\boldsymbol{x} \circ \boldsymbol{s}$ is defined by

$$(\boldsymbol{x} \circ \boldsymbol{s})_i = \begin{cases} \boldsymbol{x}^T \boldsymbol{s}, & i = 1, \\ x_1 s_i + s_1 x_i, & i = 2, \ldots, n. \end{cases}$$

Here, $\boldsymbol{e} = [1, 0, \ldots, 0]^T$. It can be checked that $\boldsymbol{x} \circ \boldsymbol{s} = \mu_k \boldsymbol{e}$ iff $s_1 = \mu_k x_1/d$ and $s_i = -\mu_k x_i/d$ for $i = 2, \ldots, n$, where d is as above, provided that $\boldsymbol{x}, \boldsymbol{s} \in C^n_2$.

Finally, if $K_1, \ldots, K_r$ all convex cones each with a Jordan product and with Jordan identities $(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_r)$, then the Jordan identity for $K_1 \times \cdots \times K_r$ is $(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_r)$ and the Jordan product is elementwise: $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r) \circ (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_r) = (\boldsymbol{x}_1 \circ \boldsymbol{s}_1, \ldots, \boldsymbol{x}_r \circ \boldsymbol{s}_r)$.

The *primal-dual interior-point method* consists of solving the system of nonlinear equations:

$$\begin{aligned} \nabla f(\boldsymbol{x}) - \boldsymbol{s} + J(\boldsymbol{x})^T \boldsymbol{\lambda} &= \boldsymbol{0}, \\ \boldsymbol{g}(\boldsymbol{x}) &= \boldsymbol{0}, \\ \boldsymbol{x} \circ \boldsymbol{s} &= \mu_k \boldsymbol{e}, \end{aligned}$$

whose variables are $(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{s})$, using Newton's method for a sequence of decreasing μ_k's, and using the converged (or approximately converged) solution $(\boldsymbol{x}_{k-1}, \boldsymbol{\lambda}_{k-1}, \boldsymbol{s}_{k-1})$ as the starting guess for the kth iteration. If Newton's method is applied directly to the above system, this yields a step called the AHO direction (Alizadeh-Haeberly-Overton). See [20] for an in-depth discussion.

In the nonconvex case, the known theorems are considerably weaker. An analysis of a primal-dual interior-point method for nonconvex second-order cone programming is presented by Yamashita and Yabe [22]. The main innovation in that work is a merit function that ensures convergence. We have experimented with a primal-dual interior point method but have not used it herein because it sometimes failed to converge to a solution and instead became trapped close to a boundary of the feasible region. The hypotheses of the Yamashita-Yabe method do not hold for the problem herein. Therefore, the method used in our solver is a primal-only method. However, we use the primal-dual formulation in the computations of energy balance detailed below.

3. Finite element discretization

As mentioned earlier, we assume a physical domain $\Omega \subset \boldsymbol{R}^{n_{\mathrm{dim}}}$ ($n_{\mathrm{dim}} = 2$ or $n_{\mathrm{dim}} = 3$), which is the closure of an open, bounded set with a piecewise smooth boundary. In this section, we describe the notation used to define a finite-element discretization of Ω and u. Assume that Ω is meshed with a

triangulation $\mathcal{T}$. The triangulation is assumed to be simplicial, although the method can be extended to meshes with hanging nodes. The n_{dim}-dimensional elements of this mesh are referred to as *bulk elements*.

As mentioned in the introduction, we further assume that Ω contains a union $\mathcal{S}$ of curves or surfaces to represent possible crack paths. For the remainder of this work, we take $\mathcal{S}$ to be the union of nonexterior bounding curves or surfaces of the bulk elements. The cohesive method inserts *interface elements* along triangle edges ($n_{\text{dim}} = 2$) or facets ($n_{\text{dim}} = 3$) for every pair of adjacent bulk elements. Let the size of $\mathcal{S}$ (number of curves or facets) be denoted n_{e}. No nodes are shared between bulk elements; instead, the topology of the mesh is determined by nodes that are shared between bulk elements and their adjacent interface elements. Let n_0 denote the total number of nodes of bulk elements. Let n_x denote the number of nodal degrees of freedom not constrained by displacement or velocity boundary conditions. Thus, $n_x \leq n_{\text{dim}} n_0$.

On each time step, an optimization problem is solved to determine the displacements at the midpoint of the time interval. For the rest of this discussion, assume the time step is fixed so that we omit the subscript for time. Let $\boldsymbol{u} \in \boldsymbol{R}^{n_{\text{dim}} \cdot n_0}$ be the vector of all nodal displacements (n_{dim} coordinate entries for each of n_0 nodes). As in [15], this value of $\boldsymbol{u}$ plays the role of the unknown at the midpoint of a time-step.

Let $\boldsymbol{x} \in \boldsymbol{R}^{n_x}$ reparameterize $\boldsymbol{u}$: $\boldsymbol{x}$ stands for the degrees of freedom associated with unconstrained nodal displacements. The relationship between $\boldsymbol{u}$ and $\boldsymbol{x}$ is as follows. There is a fixed $n_{\text{dim}} n_0 \times n_x$ matrix R such that $\boldsymbol{u} = R\boldsymbol{x} + \boldsymbol{u}_{BC}$. Here, $\boldsymbol{u}_{BC}$ is the $n_{\text{dim}} n_0$-vector that carries information about displacement boundary conditions. Note that $\boldsymbol{u}_{BC}$ will depend on the time-step index in the case of velocity boundary conditions.

The strain energy associated with a bulk element $t \in \mathcal{T}$ is given by a hyperelastic energy functional. For example, in the $n_{\text{dim}} = 2$ case, one choice for the energy is Knowles plane stress given by

$$\int_{x \in t} c_1(\text{Trace}(C(x)) + J(x)^{-2\beta}(1 + 1/\beta))\, dA,$$

where $J(x) = \det(C(x))^{(1+1/\beta)/2}$, $C = F(x)^T F(x)$ (Cauchy-Green strain), $F(x)$ is the (2-dimensional) displacement gradient, c_1, β are material constants.

The strain energy in the bulk is discretized as a function $b_0(\boldsymbol{u})$ using quadrature over elements of $\mathcal{T}$. This function $b_0(\boldsymbol{u})$ is rewritten as $b(\boldsymbol{x})$ (i.e., $b(\boldsymbol{x}) \equiv b_0(R\boldsymbol{x} + \boldsymbol{u}_{BC})$).

The momentum energy term $m_0(\boldsymbol{u})$ arising from the implicit midpoint rule is derived in [15] to be

$$m_0(\boldsymbol{u}) = \frac{2}{\Delta t^2}(\boldsymbol{u} - \boldsymbol{u}^i - \boldsymbol{v}^i \Delta t/2)^T M (\boldsymbol{u} - \boldsymbol{u}^i - \boldsymbol{v}^i \Delta t/2),$$

where Δt is the time step, M is the $n_{\text{dim}} n_0 \times n_{\text{dim}} n_0$ positive definite mass matrix, and $\boldsymbol{u}^i$ and $\boldsymbol{v}^i$ are displacement and velocity vectors from the preceding time step. Define $m(\boldsymbol{x}) = m_0(R\boldsymbol{x} + \boldsymbol{u}_{BC})$. This may be loosely regarded as the discretization of kinetic energy; see [15] for a more precise explanation. Note that $m(\boldsymbol{x})$ is a convex quadratic function of $\boldsymbol{x}$.

Next, define an interface potential to stand for the second term of (1) as in [15]. Prior to discretization of the edge/surface element, this potential for a given element edge/surface $e \in \mathcal{S}$ is

$$\int_{\eta \in \Delta} g(\delta(\psi_e(\eta)); d(\psi_e(\eta))\psi_e'(\eta)\, d\eta \tag{4}$$

where ψ_e parameterizes the edge/surface e with parameter η (a scalar for $n_{\text{dim}} = 2$; a 2-vector for $n_{\text{dim}} = 3$) which lies in a reference domain Δ, $\delta(\cdot)$ is the *effective opening displacement* as calculated from displacement jump in the element boundaries, g is the interface energy function and d is a damage variable discussed below. We follow the commonplace definition similar to Ortiz and Pandolfi [14]:

$$\delta(x) = \sqrt{[\![u_n(x)]\!]^2 + \beta^2 \|[\![u_s(x)]\!]\|^2}, \tag{5}$$

where $u_n(\cdot)$ and $u_s(\cdot)$ are the normal and tangential opening displacements at a point $x \in \Omega$ that lies on an interface. We return to these functions below. Here, β is a material constant called the *mixity parameter*.

Perhaps the simplest physically reasonable choice for g is:

$$g(\delta) = \begin{cases} l\delta + q\delta^2, & \delta \in [0, \delta_u] \\ l\delta_u + q\delta_u^2, & \delta \geq \delta_u, \end{cases} \tag{6}$$

where δ_u, the ultimate opening displacement, is a material parameter; quadratic coefficient $q = -\sigma_c/(2\delta_u)$, $l = \sigma_c$, and σ_c, the critical traction, is another material parameter. One checks that with these formulas, g is a piecewise C^1 (continuous function and first derivative) quadratic function. Its first derivative with respect to δ, g', is therefore piecewise linear and continuous. With these choices of the three parameters, $g'(0) = \sigma_c$, indicating that the initial traction (first derivative of energy with respect to δ) is σ_c. The area under the curve of the plot of $g'(\delta)$ is $G_c = \sigma_c \delta_u/2$, a material parameter called the "fracture toughness". (Note: of the three material parameters σ_c, δ_u, G_c, only two can be chosen independently as the previous equality demonstrates.)

We now generalize this formula to include a nonnegative scalar damage parameter d, initially equal to 0. The role of scalar d is to model irreversible damage to the interface. This parameter lies in $[0, \delta_u]$ and is equal to the maximum opening displacement (not exceeding δ_u) encountered over previous time values. When $d = \delta_u$, the interface has no remaining cohesion. The generalized formula is:

$$g(\delta; d) = \begin{cases} l(d)\delta, & \delta \in [0, d], \\ l(d)\delta + q(\delta - d)^2 & \delta \in [d, \delta_u], \\ l(d)\delta_u + q(\delta_u - d)^2 & \delta > \delta_u, \end{cases} \tag{7}$$

where now $q = -\sigma_c/(2\delta_u)$, $l(d) = -2(\delta_u - d)q$. When $d = 0$, the formula in the previous paragraph is recovered. See Fig. 1.

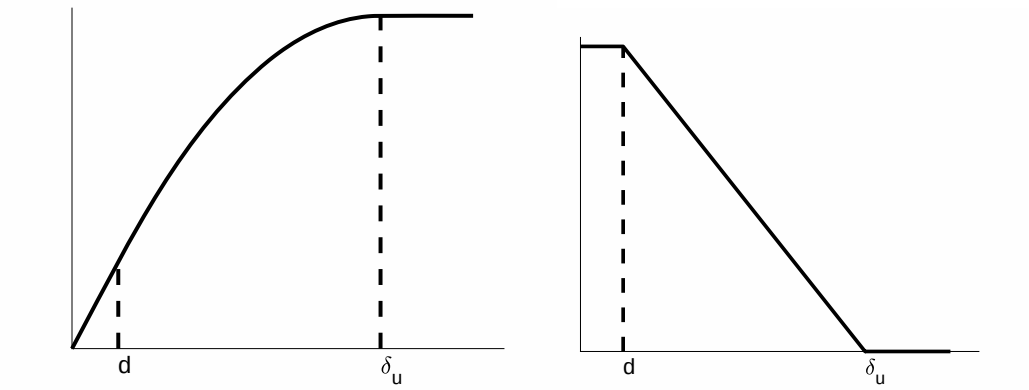

Figure 1: Plot of the function $g(\delta; d)$ (left) and $g'(\delta; d)$ (right) defined by (7).

In the finite element approximation, the integral (4) is computed for each edge ($n_{\text{dim}} = 2$) or facet ($n_{\text{dim}} = 3$) with a n_{g}-point Gauss quadrature rule. Let $n_{\text{i}} = n_{\text{e}} n_{\text{g}}$ stand for the total number of Gauss-points of interfaces. Let us introduce a new variable $\boldsymbol{s}_0 \in \boldsymbol{R}^{n_{\text{i}}}$ that represents the effective opening displacements at the Gauss points. In other words, $\boldsymbol{s}_0$ stores the vector of values of $\delta(\psi_e(\eta_\iota))$, $\iota = 1, \ldots, n_{\text{g}}$, described above at each Gauss point $\eta_1, \ldots, \eta_{n_{\text{g}}}$ of each interface e. Then the potential due to interfaces is written

$$h(\boldsymbol{s}_0; \boldsymbol{d}) = \sum_e \sum_{\iota=1}^{n_{\text{g}}} \omega_{e,\iota} g((s_0)_{e,\iota}; d_{e,\iota}); \tag{8}$$

this is the finite-element approximation to $\Phi(\llbracket u \rrbracket)$ that appears in (1). We have associated a damage variable $d_{e,\iota}$ for $e \in \mathcal{S}$ and $\iota = 1, \ldots, n_{\text{g}}$, i.e., one for each of the n_{i} interface Gauss points, that indicate the level of damage. Here, $\omega_{e,\iota}$ is the quadrature weight. This function h is separable in the entries of $\boldsymbol{s}_0$, and hence its Hessian is a diagonal matrix.

Let $\boldsymbol{s}_1 \in \boldsymbol{R}^{n_\mathrm{i}}$ denote the normal opening displacement. Let $\boldsymbol{s}_2, \ldots, \boldsymbol{s}_{n_\mathrm{dim}} \in \boldsymbol{R}^{n_\mathrm{i}}$ stand for the tangential opening displacements scaled by β. (Note that for three-dimensional problems, there is not a unique way to define a tangential coordinate system at each point on an interface. Any method for defining tangential coordinates is acceptable provided that it is applied consistently for the duration of the solution procedure.) Next, we revisit (5). First, the left-hand side is needed only at the Gauss points of the interfaces. In this case, the left-hand side is just $(s_0)_{e,\iota}$. A similar substitution may be made on the right-hand side, so this constraint is rewritten as

$$(s_0)_{e,\iota} = \sqrt{(s_1)_{e,\iota}^2 + \cdots + (s_{n_\mathrm{dim}})_{e,\iota}^2},$$

for $e = 1, \ldots, n_\mathrm{e}$, $\iota = 1, \ldots, n_\mathrm{g}$. Observe that this constraint is nondifferentiable at the origin, that is, when for some e, ι, $(s_1)_{e,\iota} = \ldots = (s_{n_\mathrm{dim}})_{e,\iota} = 0$. This nondifferentiability is fundamental to the model and is precisely the reason why it is able to capture the initially rigid interface behavior. A detailed explanation of the role of this nondifferentiability is provided in [15].

The vectors $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{n_\mathrm{dim}}$ containing the components of the opening displacements are functions of the displacements stored in $\boldsymbol{u}$. In other words, we can determine entries of $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{n_\mathrm{dim}}$ by evaluating jumps of displacements interpolated from shape functions. As mentioned earlier, we have reparameterized $\boldsymbol{u}$ by $\boldsymbol{x}$. Therefore we can define geometric functions $c_{k,e,\iota}$ such that

$$(s_k)_{e,\iota} = c_{k,e,\iota}(\boldsymbol{x}), \tag{9}$$

for $k = 1, \ldots, n_\mathrm{dim}$, $e = 1, \ldots, n_\mathrm{e}$, $\iota = 1, \ldots, n_\mathrm{g}$. As observed in [15], these functions are nonlinear because of geometric nonlinearity, namely, the normal and tangent directions depend on the current values of the displacements. In addition to geometry, the functions $c_{2,e,\iota}, \ldots, c_{n_\mathrm{dim},e,\iota}$ also have the mixity factor β encoded in them.

As mentioned in the introduction, one advantage of the interior-point formulation is the ability to handle conic convex constraints essentially for free. One special case of conic convex constraints is linear inequality contraints. Let us assume that the system has additional linear constraints of the form $E\boldsymbol{x} \geq \boldsymbol{a}$. (Later on, we will use this capability to model a simple form of a contact constraint.) Here, $E \in \boldsymbol{R}^{n_\mathrm{LI} \times n_x}$ is a known matrix and $\boldsymbol{a} \in \boldsymbol{R}^{n_\mathrm{LI}}$ is a known vector, and both may vary from one time-step to the next, and n_LI denotes the number of linear inequality constraints.

Thus, the optimization problem to solve for one time-step in the model of cohesive fracture is:

$$\begin{array}{rll} \min_{\boldsymbol{x}, \boldsymbol{s}_0, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_{n_\mathrm{dim}}} & m(\boldsymbol{x}) + b(\boldsymbol{x}) + h(\boldsymbol{s}_0; \boldsymbol{d}) + \boldsymbol{f}^T \boldsymbol{x} & \\ \text{s.t.} & (s_0)_{e,\iota} = \sqrt{(s_1)_{e,\iota}^2 + \cdots + (s_{n_\mathrm{dim}})_{e,\iota}^2} & \forall e, \iota, \\ & (s_k)_{e,\iota} = c_{k,e,\iota}(\boldsymbol{x}) & \forall k = 1, \ldots, n_\mathrm{dim}, \forall e, \forall \iota, \\ & E\boldsymbol{x} \geq \boldsymbol{a}, & \\ & (s_1)_{e,\iota} \geq 0 & \forall e, \forall \iota. \end{array} \tag{10}$$

The terms in the objective have already been discussed except for the last term, which stands for the sum of traction and body forces from (1). The first, second and third constraints were already discussed. The fourth prevents interpenetration between neighboring elements.

4. Sources of nonconvexity

The optimization problem (10) contains four source of nonconvexity, namely $b(\boldsymbol{x})$, the bulk energy, is nonconvex in the displacements for nonlinear hyperelasticity, $h(\boldsymbol{s}_0; \boldsymbol{d})$ is nonconvex in $\boldsymbol{s}_0$ due to the negative coefficient of the quadratic term in (7), the constraint $(s_0)_{e,\iota} = \sqrt{(s_1)_{e,\iota}^2 + \cdots + (s_{n_\mathrm{dim}})_{e,\iota}^2}$ defines a nonconvex, nondifferentiable surface, and the constraints $(s_i)_{e,\iota} = c_{i,e,\iota}(\boldsymbol{x})$, $i = 1, \ldots, n_\mathrm{dim}$, are nonlinear and hence define nonconvex surfaces. Of these four sources, the nonconvexity of h is the most challenging to handle, and it is also the most fundamental to the application. Convexity of $b(\boldsymbol{x})$ could be recovered by adopting a simpler mechanical model such as linear elasticity. Linearity in the constraint

$(s_k)_{e,\iota} = c_{k,e,\iota}(\boldsymbol{x})$ could be recovered by simply assuming that the normal vectors to the interfaces are determined by the initial rather than current configuration. In the next section, we explain how to convexify $(s_0)_{e,\iota} = \sqrt{(s_1)_{e,\iota}^2 + \cdots + (s_{n_{\text{dim}}})_{e,\iota}^2}$. There is, however, no apparent way to replace or approximate h with a convex function because g must have the nonconvex form similar to the form depicted in Fig. 1(left) to be physically meaningful.

Interior-point methods for nonconvex problems are considerably more delicate than for convex problems, and we were required to implement several stabilization methods in the interior-point framework to cope with the nonconvexity of h, which are as follows.

1. The interior-point method converges significantly faster if we weight the log-barrier terms for the constraints associated with interfaces by a factor proportional to their "local" length. In particular, we introduce weight $\zeta_{e,\iota} = 10^4 G_c \omega_{e,\iota}$, where $\omega_{e,\iota}$ is the quadrature weight in (8). Using a weight proportional to $\omega_{e,\iota}$ is physically natural because it means, for example, that the contribution to the barrier function from an interface is invariant (up to discretization error) if the interface is subdivided into smaller pieces. Making the weight proportional to G_c, the fracture toughness, is also natural since the other terms in the objective function stand for work or energy quantities. Weighting is not necessary (and is typically not even considered) in the case of convex interior-point methods.

2. On intermediate stages of the interior-point method, the interfaces are favored to open by the log-barrier term associated with the constraint $(s_1)_{e,\iota} \geq 0$ when μ is large. Without extra measures, they can open by more than δ_u, in which case their traction is 0 and they no longer hold the body together. Their traction is not recovered as μ is decreased due to the nonconvexity of h. For this reason, a quadratic regularization term is added to the cohesive traction. This term has the form

$$(5 \cdot 10^5)\alpha\omega_{e,\iota} \max(1 - d_{e,\iota}/\delta_u, 8 \cdot 10^{-6})(s_0)_{e,\iota}^2,$$

   where α is a scalar depending on the interior-point parameter μ (see exact formulas below), $\omega_{e,\iota}$ is the quadrature weight in (8), $d_{e,\iota}$ is the damage value, $(s_0)_{e,\iota}$ is the (unknown) effective opening displacement of Gauss point ι of the interface e. From now on, we denote:

$$h_\alpha(\boldsymbol{s}_0; \boldsymbol{d}) = h(\boldsymbol{s}_0; \boldsymbol{d}) + \sum_{e,\iota} (5 \cdot 10^5) k \omega_{e,\iota} \max(1 - d_{e,\iota}/\delta_u, 0.000008)(s_0)_{e,\iota}^2, \tag{11}$$

   where α will be selected later.

3. The Newton step associated with the interior-point step is not always well defined because the Hessian may not be positive definite. To address this problem, we compute for each μ a Hessian of a regularized problem. Then if a non-positive definite Hessian is encountered, we add a multiple of the regularizing Hessian to the true Hessian until positive definiteness is attained. This is reminiscent of a trust-region method, except we use a true Hessian from an artificial problem instead of the identity matrix. Details are provided below.

The net effect of these stabilization techniques is that the method is reasonably fast and robust on all the computational experiments tried so far.

5. Interior-point method for fracture

A key modification to (10) that makes it amenable to an interior-point method is to replace the first equality constraint with an inequality constraint:

$$\begin{array}{rll} \min_{\boldsymbol{x},\boldsymbol{s}} & m(\boldsymbol{x}) + b(\boldsymbol{x}) + h(\boldsymbol{s}_0; \boldsymbol{d}) + \boldsymbol{f}^T \boldsymbol{x} & \\ \text{s.t.} & (s_0)_{e,\iota} \geq \sqrt{(s_1)_{e,\iota}^2 + \cdots + (s_{n_{\text{dim}}})_{e,\iota}^2} & \forall e, \forall \iota, \\ & (s_k)_{e,\iota} = c_{k,e,\iota}(\boldsymbol{x}) & \forall k = 1, \ldots, n_{\text{dim}}, \forall e, \forall \iota, \\ & E\boldsymbol{x} \geq \boldsymbol{a}, & \\ & (s_1)_{e,\iota} \geq 0 & \forall e, \iota. \end{array} \tag{12}$$

Replacing the equality by an inequality constraint does not change the optimizer because h is a nondecreasing function of $\boldsymbol{s}_0$. (In fact, it is perturbed to a strictly increasing function as described in the previous section.) This implies that the optimal solution to (12) satisfies $(s_0)_{e,\iota} = \sqrt{(s_1)_{e,\iota}^2 + \cdots + (s_{n_{\mathrm{dim}}})_{e,\iota}^2}$.

The benefit of this change is that an equality constraint of the form $(s_0)_{e,\iota} = \|(\boldsymbol{s}_{1:n_{\mathrm{dim}}})_{e,\iota}\|$ defines a nonconvex set with a complicated (nonmanifold) structure, whereas the constraint $(s_0)_{e,\iota} \geq \|(\boldsymbol{s}_{1:n_{\mathrm{dim}}})_{e,\iota}\|$ defines a convex set $C_2^{n_{\mathrm{dim}}+1}$, the second-order cone. Here, $(\boldsymbol{s}_{1:n_{\mathrm{dim}}})_{e,\iota}$ is the vector in $\boldsymbol{R}^{n_{\mathrm{dim}}}$ whose entries are $((s_1)_{e,\iota}, \ldots, (s_{n_{\mathrm{dim}}})_{e,\iota})$; similarly, $(\boldsymbol{s}_{0:n_{\mathrm{dim}}})_{e,\iota}$ is notation for the vector in $\boldsymbol{R}^{n_{\mathrm{dim}}+1}$ that includes the indicated entry of $\boldsymbol{s}_0$.

As discussed in Section 2, the primal-only interior-point method replaces the inequality constraints in the preceding formulation with log-barrier terms. The parameter $\mu > 0$ starts at a large value and decreases to close to zero. This leads to the following formulation:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{s}} \quad & m(\boldsymbol{x}) + b(\boldsymbol{x}) + h_\mu(\boldsymbol{s}_0; \boldsymbol{d}) + \boldsymbol{f}^T\boldsymbol{x} + \mu\phi_{\mathrm{NNO}}(E\boldsymbol{x} - \boldsymbol{a}) \\ & \quad + \mu \textstyle\sum_{e=1}^{n_{\mathrm{e}}} \sum_{\iota=1}^{n_{\mathrm{g}}} \zeta_{e,\iota} \left[\phi_{\mathrm{SOC}}((\boldsymbol{s}_{0:n_{\mathrm{dim}}})_{e,\iota}) + \phi_{\mathrm{NNO}}((s_1)_{e,\iota})\right] \\ \text{s.t.} \quad & (s_k)_{e,\iota} = c_{k,e,\iota}(\boldsymbol{x}) \quad \forall k = 1, \ldots, n_{\mathrm{dim}}, \forall e, \forall \iota. \end{aligned} \tag{13}$$

As mentioned above, weight $\zeta_{e,\iota}$ is the norm of the derivative of the interface shape function (i.e., the "local length" of the interface) at Gauss point e, ι. Also, we have set α appearing in (11) equal to μ. In other words, the weight on the quadratic regularizing term gets small as μ tends to 0.

This formulation still contains equality constraints, but they are easily eliminated via substitution. In particular, we substitute $c_{k,e,\iota}(\boldsymbol{x})$ for $(s_k)_{e,\iota}$ (as in (9)) thus eliminating $(s_k)_{e,\iota}$, $k = 1, \ldots, n_{\mathrm{dim}}$, $e = 1, \ldots, n_{\mathrm{e}}$, $\iota = 1, \ldots, n_{\mathrm{g}}$. This elimination leaves the following unconstrained problem:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{s}_0} \quad & b(\boldsymbol{x}) + m(\boldsymbol{x}) + h_\mu(\boldsymbol{s}_0; \boldsymbol{d}) + \boldsymbol{f}^T\boldsymbol{x} + \mu\phi_{\mathrm{NNO}}(E\boldsymbol{x} - \boldsymbol{a}) \\ & \quad + \mu \textstyle\sum_{e=1}^{n_{\mathrm{e}}} \sum_{\iota=1}^{n_{\mathrm{g}}} \zeta_{e,\iota} \Big[\phi_{\mathrm{SOC}}((s_0)_{e,\iota}, \boldsymbol{c}_{1:n_{\mathrm{dim}},e,\iota}(\boldsymbol{x})) \\ & \quad + \phi_{\mathrm{NNO}}(c_{1,e,\iota}(\boldsymbol{x}))\Big] \end{aligned} \tag{14}$$

The interior-point code solves (14) via Newton's method. The code uses a regularized Newton's method in which a regularizing matrix may be added to the Hessian. The regularizing matrix is the Hessian computed in an earlier artificial iteration in which the interior-point method is run to completion with $h_{\mu_{\mathrm{init}}}$ in place of h_μ. The regularizing matrix is added with an increasing coefficient until a valid step is found. 'Valid' means that the step yields a feasible point, that sufficient decrease occurs in the function value, and that the norm of the gradient does not increase by more than a constant factor. 'Feasible' in this context means $c_{1,e,\iota}(\boldsymbol{x}) > 0$, $(s_0)_{e,\iota} > (c_{1,e,\iota,}(\boldsymbol{x})^2 + \cdots + c_{n_{\mathrm{dim}},e,\iota}(\boldsymbol{x})^2)^{1/2}$ for all e, ι, and $E\boldsymbol{x} > \boldsymbol{a}$. These inequalities imply that all the barrier functions are well-defined. The test for feasibility also checks that no element is inverted, i.e., it checks that all determinants of displacement gradient matrices are positive at Gauss points of the bulk elements.

6. Attaining feasibility using Phase I

An issue to address is that interior-point methods for optimization require an interior starting point. There are so-called "infeasible" interior-point methods that ease this restriction, but the theory for such methods in the case of nonconvex problems is not well developed, and in practice they can be difficult to use. A natural choice for initialization is the converged solution for the previous time step, but such a solution may violate the inequality constraints such as contact. In addition, for moving displacement boundary conditions (i.e., velocity boundary conditions), the boundary nodes must be displaced each iteration to new positions to attain feasibility. Depending on the magnitude of the velocity, this can cause elements along the boundary to become inverted or nearly inverted, which makes the bulk material model behave poorly to the extent that recovering a noninverted shape is unattainable with Newton's method.

One solution to the problem described in the last paragraph is to take small load steps, thus limiting the degree of misshapenness among boundary elements. The drawback of this technique is that it makes the

time-step dependent on the mesh size because a finer mesh implies that a smaller boundary distortion can be tolerated. Recall that the method proposed herein is an implicit method. A big advantage of implicit methods is exactly that the time step and mesh size can be chosen independently. Thus, coping with moving boundaries by subdividing time-steps undermines one of the main benefits of implicit methods.

An alternative approach, adopted here, is to start with a so-called "Phase I" initialization. During Phase I, the optimization problem is modified with a new variable t constrained $t \geq 0$ and a "big M" term in the objective. A preliminary version of the modified problem is as follows.

$$\begin{aligned}\min_{\boldsymbol{x},\boldsymbol{s}_0,t} \quad & b(\boldsymbol{x}) + m(\boldsymbol{x}) + h_{\mu_{\text{init}}\sqrt{M}}(\boldsymbol{s}_0;\boldsymbol{d}) + \boldsymbol{f}^T\boldsymbol{x} + Mt + \mu_{\text{init}}\phi_{\text{NNO}}(E\boldsymbol{x} + t\boldsymbol{e} - \boldsymbol{a}) + \mu_{\text{init}}\phi_{\text{NNO}}(t) \\ & + \mu_{\text{init}} \sum_{e=1}^{n_{\text{e}}} \sum_{\iota=1}^{n_{\text{g}}} \zeta_{e,\iota} \Big[\phi_{\text{SOC}}((s_0)_{e,\iota}, \boldsymbol{c}_{1:n_{\text{dim}},e,\iota}(\boldsymbol{x})) \\ & \quad + \phi_{\text{NNO}}(c_{1,e,\iota}(\boldsymbol{x}) + t\boldsymbol{e}) \Big]\end{aligned} \tag{15}$$

Here, $\boldsymbol{e}$ is the vector of all 1's. Note that during Phase I, μ is held at its initial value.

The point of (15) is that for t sufficiently large, existence of a feasible solution is assured. The optimizer of (15), on the other hand, will have a small $t > 0$ due to the objective term Mt. After a solution is found to (15), we check whether it is feasible for (14), and if not, we re-solve (15) with M increased by a factor of 8 (initially, $M = 64$). A larger value of M means a smaller value of $t > 0$ at the optimizer. The parameter α appearing in h_α in (11) is $\mu_{\text{init}}\sqrt{M}$ in order to ensure that the quadratic penalty term in h is not swamped by the Mt term in the objective.

Now we modify (15) to handle moving boundary conditions as follows. Recall that the vector of all degrees of freedom (DOFs) $\boldsymbol{u}$ is parameterized by $\boldsymbol{u} = R\boldsymbol{x} + \boldsymbol{u}_{\text{BC}}$, where $\boldsymbol{u}_{\text{BC}}$ carries the prescribed displacements and R is a fixed matrix. An alternative way to write the boundary constraints is $B\boldsymbol{u} = \boldsymbol{b}$, i.e., linear equations that must be satisfied by $\boldsymbol{u}$. The transformation from $(R, \boldsymbol{u}_{\text{BC}})$ to $(B, \boldsymbol{b})$ can be carried out using standard numerical linear algebra such as QR factorization. Note that both $\boldsymbol{u}_{\text{BC}}$ and $\boldsymbol{b}$ may depend on the time step τ, whereas R and B are fixed throughout the computation.

In Phase I, we let the vector of unknown displacements be $\boldsymbol{u}$ rather than $\boldsymbol{x}$. Instead of equality constraints $B\boldsymbol{u} = \boldsymbol{b}$, we introduce two inequalities $B\boldsymbol{u} + t\boldsymbol{e} \geq \boldsymbol{b}$ and $B\boldsymbol{u} - t\boldsymbol{e} \leq \boldsymbol{b}$, where t is the Phase-I artificial variable. In turn, these two constraints are replaced by additional terms in the barrier objective function of the form $\mu\phi_{NNO}(B\boldsymbol{u}+t\boldsymbol{e}-\boldsymbol{b})$ and $\mu\phi_{NNO}(-B\boldsymbol{u}+t\boldsymbol{e}+\boldsymbol{b})$. Thus, the problem is reformulated as:

$$\begin{aligned}\min_{\boldsymbol{u},\boldsymbol{s}_0,t} \quad & b_0(\boldsymbol{u}) + m_0(\boldsymbol{u}) + h_{\mu_{\text{init}}\sqrt{M}}(\boldsymbol{s}_0;\boldsymbol{d}) + \boldsymbol{f}_0^T\boldsymbol{u} + Mt + \mu_{\text{init}}\phi_{\text{NNO}}(E_0\boldsymbol{u} + t\boldsymbol{e} - \boldsymbol{a}_0) + \mu_{\text{init}}\phi_{\text{NNO}}(t) \\ & + \mu_{\text{init}}\phi_{\text{NNO}}(B\boldsymbol{u} + t\boldsymbol{e} - \boldsymbol{b}) + \mu_{\text{init}}\phi_{\text{NNO}}(-B\boldsymbol{u} + t\boldsymbol{e} + \boldsymbol{b}) \\ & + \mu_{\text{init}} \sum_{e=1}^{n_{\text{e}}} \sum_{\iota=1}^{n_{\text{g}}} \zeta_{e,\iota} \Big[\phi_{\text{SOC}}((s_0)_{e,\iota}, \boldsymbol{c}_{0;1:n_{\text{dim}},e,\iota}(\boldsymbol{u})) \\ & \quad + \phi_{\text{NNO}}(c_{0;1,e,\iota}(\boldsymbol{u}) + t\boldsymbol{e}) \Big]\end{aligned} \tag{16}$$

Here, we recall the notation introduced earlier that $b_0(R\boldsymbol{x}+\boldsymbol{u}_{\text{BC}}) \equiv b(\boldsymbol{x})$ and similarly for $m_0(\cdot)$, $\boldsymbol{f}_0, E_0, \boldsymbol{a}_0, \boldsymbol{c}_0(\cdot)$.

7. Computational procedure

In this section, we provide further details of the computational procedure of the cohesive solver. As mentioned in the last section, during Phase I, the displacement boundary conditions are enforced as inequalities, and therefore the vector of unknowns for the displacements in (16) is $\boldsymbol{u}$ rather than $\boldsymbol{x}$. Let Π_τ be the linear projection that maps $\boldsymbol{u}$ to $\boldsymbol{x}$, i.e., $\Pi_\tau(\boldsymbol{u}) = \text{argmin}_{\boldsymbol{x}} \|(R\boldsymbol{x} + \boldsymbol{u}_{BC,\tau}) - \boldsymbol{u}\|$, where τ is the time step. The feasibility test that terminates Phase I that was described in the previous section in more detail is: apply Π_τ to $\boldsymbol{u}$ that solves (16) to make sure that the vector $\boldsymbol{x}$ thus obtained is feasible for the main phase.

For the purpose of notation in the solvers, let $\boldsymbol{\xi}$ denote the concatenation $(\boldsymbol{x}, \boldsymbol{s}_0)$, the variables of the interior method (14), or $\bar{\boldsymbol{\xi}} = (\boldsymbol{u}, \boldsymbol{s}_0, t)$ in the case of Phase I when (16) is solved.

In the algorithms that follow, these variables $\boldsymbol{\xi}$ and $\bar{\boldsymbol{\xi}}$ are not meant to stand for new independent program variables but merely notational shorthand. For example, if $\boldsymbol{x}$ is updated in a code that follows, then $\boldsymbol{\xi}$ is also updated implicitly since $\boldsymbol{\xi}$ contains $\boldsymbol{x}$ as a subvector.

We make the following observation: given a value of $\boldsymbol{x}$ or $\boldsymbol{u}$, it is possible to efficiently compute the optimal extension of $\boldsymbol{x}$ to $\boldsymbol{\xi}$ or $\boldsymbol{u}$ to $\bar{\boldsymbol{\xi}}$, where "optimal" in this context means minimizing the relevant objective function f. One observes from (12) that once $\boldsymbol{x}$ is specified, the remaining variables $(s_0)_{e,\iota}$, $e = 1, \ldots, n_{\mathrm{e}}$, $\iota = 1, \ldots, n_{\mathrm{g}}$ (and t in the case of $\bar{\boldsymbol{\xi}}$) are decoupled and may be optimized individually using a univariate procedure (e.g., bisection). Let us denote these optimal values as $\boldsymbol{\xi}^*(\boldsymbol{x})$, $\bar{\boldsymbol{\xi}}^*(\boldsymbol{u})$.

The top-level procedure is described in Fig. 2. Every third time step, the algorithm computes matrices H_{μ_i} for $i = 1, \ldots, n_\mu$, which are positive definite matrix used to regularize the Hessian.

The variables maintained in the main loop from one time step to the next are $\boldsymbol{u}$ and $\boldsymbol{d}$, which are superscripted with the time step index τ. As discussed earlier, $\boldsymbol{u}$ (or $\boldsymbol{x}$) encodes the displacements while $\boldsymbol{d}$ is the damage state, which is updated from the displacements computed on each step.

From initial conditions determine initial guess for $\boldsymbol{u}^0$.
$\boldsymbol{d}^0 := \boldsymbol{0}$.
$\{H_\mu\}_\mu := 0 \, \forall \mu; \quad H^{\mathrm{P1}} := 0$.
for $\tau := 1, \ldots, n_{\mathrm{step}}$
  if $\mathrm{rem}(\tau, 3) == 1$
    $H^{\mathrm{P1}}, \{H_\mu\}_\mu = \texttt{solver}(\tau, \boldsymbol{u}^0, \boldsymbol{d}^0, H^{\mathrm{P1}}, \{H_\mu\}_\mu, \mathit{PREPROCESS})$
  end
  $\boldsymbol{u}^\tau := \texttt{solver}(\tau, \boldsymbol{u}^{\tau-1}, \boldsymbol{d}, H^{\mathrm{P1}}, \{H_\mu\}_\mu, \mathit{ORDINARY})$
  Compute $\boldsymbol{d}^\tau$ from $\boldsymbol{d}^{\tau-1}, \boldsymbol{u}^\tau$.
end for

Figure 2: Top-level procedure.

The sequence of μ values used in the interior-point methods are fixed in advance as a geometrically decreasing sequence: $\mu_i = \mu_{\mathrm{init}} \rho_\mu^{i-1}$ for $i = 1, 2, \ldots, n_\mu$. The choice of parameters used is $\mu_{\mathrm{init}} = 5 \cdot 10^{-5}$, $\rho_\mu = 0.125$, and $n_\mu = 6$. This means that the ultimate value is $\mu_{n_\mu} \approx 1.5 \cdot 10^{-9}$.

The procedure `solver` to solve one time step using the interior-point method is detailed in Fig. 3.

Note that the bulk strain-energy function $b(\boldsymbol{x})$ in general depends on the time step index. This is because this function $b(\cdot)$ encodes all of the boundary and loading conditions. The momentum energy term $m(\boldsymbol{x})$ also varies with the time step because it incorporates values of the nodal velocities from previous time steps.

The interior-point inner loop, which appears in Fig. 4, defines a parameter λ indicating the degree to which the Hessian of (14) is regularized. Initially $\lambda = 0$, meaning no regularization. Later, it is first incremented to a small positive value ($\lambda_{\min} = 2^{-40}$ in the present code) and then doubled until the inner iteration succeeds. Thus, the purpose of the inner loop is to determine λ. The inner loop finishes when the modified Hessian G is positive definite, when the resulting step decreases the objective f, and when the new gradient (transformed by G^{-1}) is not much larger than the old gradient. The last test is necessary because we found occasionally that the gradient can tend to ∞ even as the function decreases due to the nonconvexity and sharp derivatives of the barrier functions.

Subsequent outer iterations diminish λ by a factor of 2 per outer iteration until it decreases $\lambda_{\min}$ and then to 0 after that. (But the inner iteration may raise it again.) The outer iteration terminates when the Newton step to update $\boldsymbol{\xi}$ is sufficiently small.

8. Computation of energy balance

In the computational experiments in the next section, we report on energy balance in two computational experiments. In this section, we describe the terms that enter into the energy balance. The energy balances are computed at half-steps between the main time steps, since this is where the displacements

FUNCTION `solver`$(\tau, \boldsymbol{u}, \boldsymbol{d}, H^{\mathrm{P1}}, \{H_\mu\}_\mu, \mathit{flag1})$
Determine functions $b(\cdot)$, $m(\cdot)$, $b_0(\cdot)$, $m_0(\cdot)$ for time step τ.
$\bar{\boldsymbol{\xi}} := \bar{\boldsymbol{\xi}}^*(\boldsymbol{u})$
$M := 64$; $\mu := \mu_1$.
/* Phase 1 to find feasible $\boldsymbol{x}$ */
loop
  Let $f(\cdot)$ be the objective function of (16).
  $(\boldsymbol{u}, \boldsymbol{s}_0, t) :=$ `minimize`$(f, \bar{\boldsymbol{\xi}}, H^{\mathrm{P1}})$.
  $\boldsymbol{x} := \Pi_\tau(\boldsymbol{u})$.
  If $(\boldsymbol{x}, \boldsymbol{s}_0)$ feasible for (14)
    break
  end
  $M := 8M$.
end loop
$\boldsymbol{\xi} := (\boldsymbol{x}, \boldsymbol{s}_0)$.
if *flag1* ==*PREPROCESS*
  $H^{\mathrm{P1}} = \nabla^2 f(\boldsymbol{u})$.
end if
/* Phase 2 to minimize (13) */
for each $\mu := \mu_1, \mu_2, \ldots, \mu_{n_\mu}$
  $k := \begin{cases} \mu_1 & \text{if } \mathit{flag1} == \mathit{PREPROCESS} \\ \mu & \text{if } \mathit{flag1} == \mathit{ORDINARY}. \end{cases}$
  Let $f(\cdot)$ be the objective function of (13).
  $\boldsymbol{\xi} :=$ `minimize`$(f, \boldsymbol{\xi}, H_\mu)$.
  if *flag1*==*PREPROCESS*
    $H_\mu := \nabla^2 f(\boldsymbol{\xi})$
  end if
end for
if *flag1* ==*PREPROCESS*
  return $H^{\mathrm{P1}}, \{H_\mu\}_\mu$
else
  return $R\boldsymbol{x} + \boldsymbol{u}_{\mathrm{BC}}$.
end if

Figure 3: Solver for (12).

```
FUNCTION minimize(f, ξ, H̄)
λ := 0
loop
    λ := { 0,      λ ≤ λ_min,
         { λ/2,    else
    loop
        ν := ‖∇²f(ξ)‖₁
        G := ∇²f(ξ) + λ(H̄ + 10⁻³νI)
        if G is positive definite
            Δξ := −G⁻¹∇f(ξ)
            ξ^test := ξ + Δξ
            if f(ξ^test) < f(ξ) and ‖G⁻¹∇f(ξ^test)‖ ≤ 1.5‖G⁻¹∇f(ξ)‖
                break
            end if
        end if
        λ := { λ_min,  λ = 0
             { 2λ,     else
    end loop
    if ‖Δξ‖ ≤ 5 · 10⁻⁷ and λ ≤ 2⁻²⁴
        return ξ
    end if
    ξ := ξ + Δξ
end loop
```

Figure 4: Primal-only procedure

are computed by the implicit midpoint rule. In this context, we use energy balance mainly as a means to validate the method, but we note that energy balance computations for initially rigid cohesive fracture in the previous literature, e.g., Molinari et al. [12], have been used to derive novel results on convergence behavior.

Kinetic energy is evaluated using quadrature of

$$\frac{1}{2}\int_\Omega \rho \dot{u}^2 \, dV,$$

The value for $\dot{u}$ for use in this integral is taken to be the midpoint of the velocities evaluated at two consecutive time-steps. Strain energy is evaluated using quadrature on the first term of (1) at the time-step midpoint using the displacements at that midpoint.

For a conservative model, cohesive energy would be evaluated using quadrature on the second term of (1). However, recall that we have introduced damage variable $\boldsymbol{d}$, so the calculation is more complicated and is described in the caption of Fig. 5.

The stored energies just described must be balanced against the work done on the models. The first source of work is from the traction and body forces (the last two terms of (1)). Since our examples do not involve either traction or body forces, we omit a detailed discussion of these terms, but their computation is relatively straightforward.

The work done by moving displacement (i.e., velocity) boundary conditions is computed as follows. The objective function $f(\boldsymbol{u}; \boldsymbol{s}_0)$ is written down as in (14), except that all functions are written in terms of $\boldsymbol{u}$ (the vector of all DOFs) instead of $\boldsymbol{x}$ (the vector of unconstrained DOFs). Next, the gradient with respect to $\boldsymbol{u}$ is computed at the minimizer for the final value of μ. The gradient entries corresponding to the constrained DOF's will in general not vanish because the objective function is not minimized with respect to them. In fact, the gradient entries are exactly the reaction forces for those DOFs. Therefore, the work due to displacement constraints during a single time-step, that is, from one half-time-step to

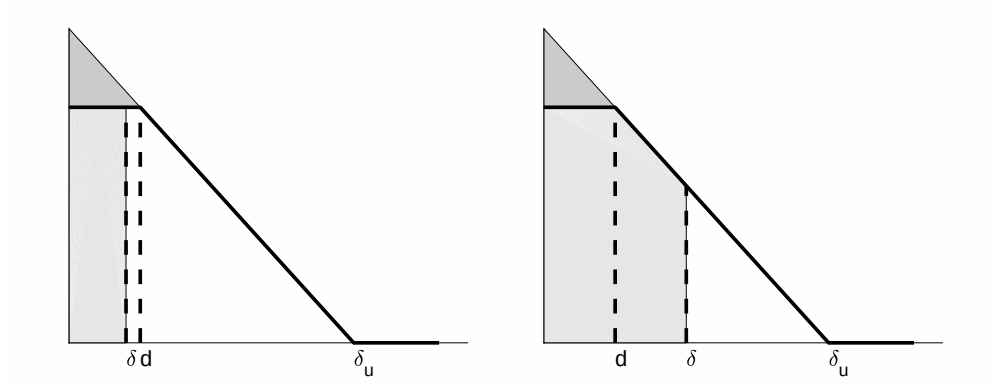

Figure 5: Plot of the computation of cohesive energy. The figure in the left considers the case $\delta < d$, where δ is the current opening displacement and d is the damage value at an interface Gauss point. The figure on the right considers $\delta > d$. Note that in a continuous-time model, one would never have $\delta > d$. In our computations, however, the update to d lags one time-step behind the computation of δ. The lightly shaded area in both figures indicates the energy that can be recovered if the interface unloads. The darkly shaded area indicates dissipated energy due to irreversibility. The total cohesive energy associated with the Gauss point is therefore the sum of the two areas.

the next, is evaluated as follows. One computes the inner product of the time-average of these forces (average between the current half-time-step and the previous) and the distance traveled by each such DOF between the current and previous half-time-step.

Finally, work done by contact boundary conditions is also obtained as an inner product. One multiplies the contribution to the force balance from the derivative of the contact barrier term that appears in $\nabla_{\boldsymbol{u}} f(\boldsymbol{u}; \boldsymbol{s}_0)$ by the distance traveled. Time-averages are used as in the last paragraph. Note that although the coefficient μ in front of the barrier term may vanish, the corresponding term of the gradient does not vanish as $\mu \to 0$ but instead tends to a constant value; this is a well-known aspect of interior-point theory.

9. Computational experiments

In this section we describe two computational experiments. The first involves impact of a metal striker on a compact compression specimen (CCS). This application demonstrates the ease in which convex constraints can be included in the computation. In particular, we model contact between the striker and the specimen via inequalities between x-coordinates of matching nodes at the ends of the strikers and the specimen. Refer to Fig. 6 for the correspondence.

The CCS is made of PMMA of size height $\times$ width $=$ 51mm $\times$ 46mm. Its initially undeformed configuration is depicted in Fig. 6. It is struck on the left boundary by a striker moving in the positive x direction. This computation simulates an experiment by Rittel and Maigre [19]. Note that, although the elasticity properties are reported as E and ν, in fact the nonlinear hyperelastic model used in the CCS is Knowles and Sternberg's [9] plane stress, whose material parameters are c_1 and β (unrelated to the cohesive mixity parameter). These are obtained from E and ν according to the formulas $c_1 = E/(4(1+\nu))$, $\beta = \nu/(1-2\nu)$.

No boundary conditions are applied, i.e., all boundaries are unconstrained and traction-free. The striker is initially moving at uniform velocity of 12 m/s in the positive x direction, while the CCS is initially at zero velocity. The initial displacement fields are all zeros. The initial mesh of the CCS contains 1765 nodes and 820 element (quadratic triangles). After duplication of nodes (recall that in our finite element model, each triangle "owns" its own nodes, and interface elements are inserted to topologically connect neighboring triangles), the number of nodes is $6 \cdot 820 = 4920$. The mesh near the fracture zone is a "pinwheel" mesh [6] transformed by a nonlinear coordinate transformation, as depicted in Figure 6. Pinwheel meshes have the property that in the limit of mesh refinement, all possible crack orientations are represented in the mesh, so they are well suited for computations in which determining

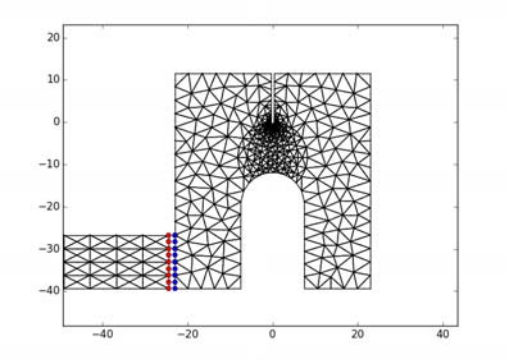

Figure 6: Initial geometry and mesh of the striker and CCS specimen. Contact boundary conditions are applied between the nodes colored red and corresponding nodes colored blue. For clarity of the figure, the width of the initial gap between the striker and CCS has been exaggerated compared to the simulation.

the crack path is part of the problem. (Other techniques have been proposed for representing many possible orientations in a mesh, e.g., adaptive splitting of polygonal elements by Leon et al. [10].) The striker is modeled as with quadratic triangles with 1403 nodes and 656 element. Thus, the final model contains 6323 nodes and 1476 quadratic triangles. No interface elements are present in the striker. The material model for the striker is isotropic linear elasticity.

The time step is 3 microseconds. The simulated crack path after 32 time steps (96 simulated microseconds)—refer to Fig. 7 and Fig. 8—is similar to experimental results in [19].

The test was run on a 2.6GHz Intel Xeon E5-2690 running Linux. The algorithm was coded in the Julia [2] programming language, version 0.6.2. The code is not parallelized yet. The computation time for 32 time steps was 2 hours.

Property	Symbol	PMMA	Steel	Asphalt concrete
Young modulus	E	5.76 GPa	200 GPa	38 GPa
Poisson ratio	ν	0.42	0.3	0.18
Density	ρ	1180 kg/m^3	8050 kg/m^3	—
Critical traction	σ_c	105 MPa	—	3 MPa
Mixity parameter	β	2.0	—	4.0
Fracture toughness	G_c	352 Pa·m	—	69 Pa·m

Table 1: Material properties used in computational experiments.

We also tracked energy balance in the two computational experiments. The technique used to measure energy balance was described in Section 8. Because the optimization problem is solved at the midpoint of timesteps in the implicit midpoint rule used herein, we evaluate the energy balance at time-step midpoints. However, not all the variables are evaluated at time-step midpoints, so interpolations must be used. Therefore, we would not expect exact energy balance because the different terms involve different approximation assumptions.

The second computational test exhibits the performance of the method in the quasi-static (slow loading) regime. In this problem, taken from Galvez et al. [5], a beam of concrete 15 cm × 67.5 cm, plane stress, with an initial centered slit is subjected to moving displacement boundary concentrated at an off-center point. The problem set-up is described in more detail in Fig. 10.

A depiction of the configuration after 25 load steps is shown in Fig. 11. The number of elements is 3168 and the number of nodes (after duplication) is 19,008. This computation required 15 hours. (The amount would be greatly reduced if we had inserted interface elements only in the zone where crack

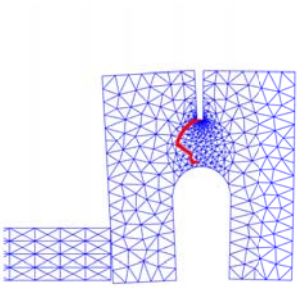

Figure 7: Damaged interfaces after 32 time steps (96 microseconds of simulated time).

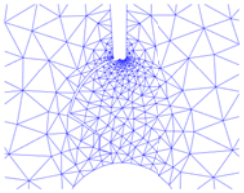

Figure 8: Closeup of Fig. 7.

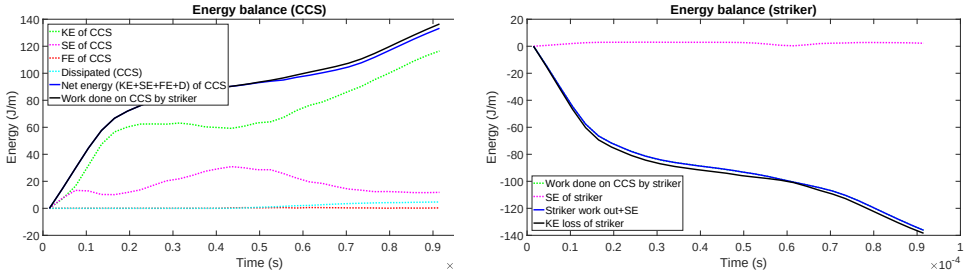

Figure 9: Eneergy balances for the CCS experiment; "KE" stands for kinetic energy, "SE" stands for strain energy, and "FE" stands for fracture energy. Refer to Section 8 for an explanation of how the energy and work contributions were computed. In each figure, if energy were exactly conserved, the blue and black curves would coincide.

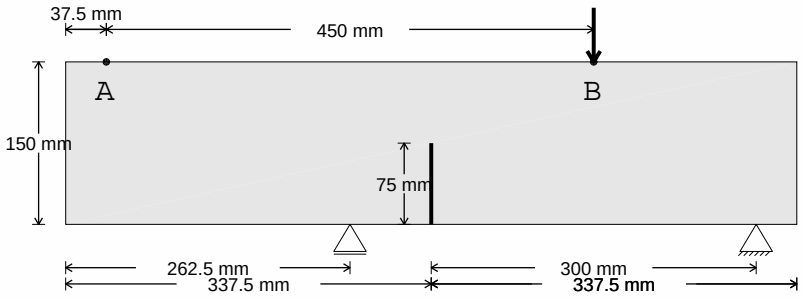

Figure 10: Quasi-static test of concrete beam. The beam is supported at two asymmetric points of its base and has an initial vertical slit. The point B is loaded quasi-statically. The point A is referred to in Fig. 13 below.

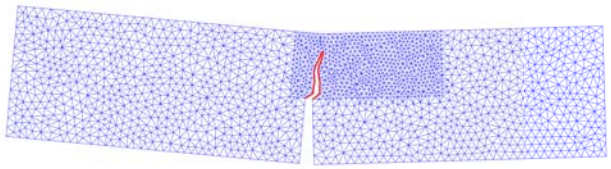

Figure 11: Final configuration of concrete beam after 25 steps. The displacements are exaggerated by a factor of 50 for better visualization.

propagation is known to occur i.e., above and to the right of the initial slit, whereas in fact our mesh has cohesive interfaces at every interelement boundary.) The crack path roughly matches the experiment in [5], although we were not trying to accurately reproduce the path in this experiment because we did not use a pinwheel or other special mesh.

An energy balance was also computed for the Galvez experiment; the results are reported in Fig. 12. In addition, a load-deflection curve was plotted since this data is available experimentally. The result of this computation appears in Fig. 13.

It is interesting to compare the energy balance in the CCS experiment compared to the Galvez experiment. In the CCS experiment, most of the work done by the striker becomes kinetic energy of the CCS. Part becomes strain energy, but the strain energy is partly released as it is transformed first to cohesive energy and the dissipated. In the case of the Galvez problem, there is no kinetic energy (the problem is quasi-static), so the energy of the load first builds up the strain energy, which is then released as cohesive energy, and then two steps later the fracture energy is dissipated.

10. Conclusions

An interior-point method for initially rigid cohesive fracture is proposed. A key technical step to make this method possible is the replacement of an equation relating the effective opening displacement to the coordinate entries of the opening displacement by an inequality constraint. A computational test shows that the method is practical and can easily encompass additional conic inequality constraints.

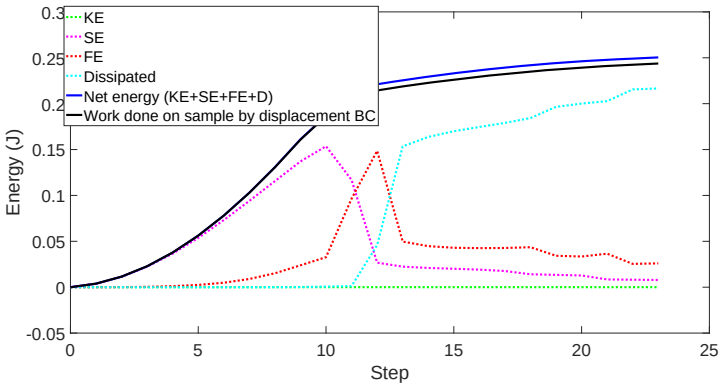

Figure 12: Energy balance for Galvez experiment. Refer to the caption of Fig. 9 for further information. Thickness has been normalized to 5cm to correspond to the experiment.

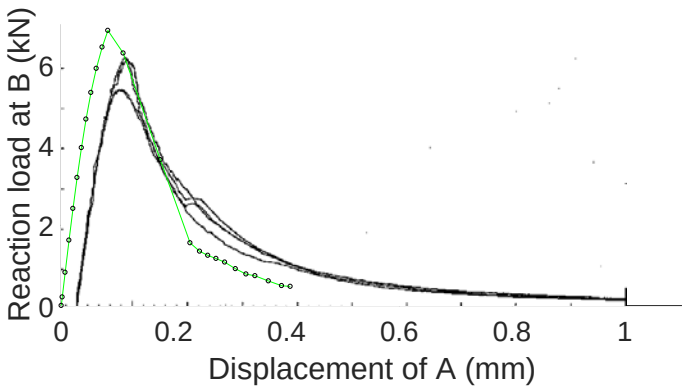

Figure 13: A plot of the load (reaction force) at point B versus the vertical deflection of point A. The experimental envelope from Galvez et al. [5] is represented by gray curves. Thickness has been normalized to 5cm to correspond to the experiment.

References

[1] G. I. Barenblatt. The mathematical theory of equilibrium cracks in brittle fracture. *Advances in Applied Mechanics*, 7:55–129, 1962.

[2] J. Bezanson, A. Edelman, S. Karpinski, and V.B. Shah. Julia: A fresh approach to numerical computing. *SIAM Rev.*, 59(1):65–98, 2017.

[3] B. Bourdin, G.A. Francfort, and J.-J. Marigo. The variational approach to fracture. *J Elasticity*, 91(1-3):5–148, 2008.

[4] D. S. Dugdale. Yielding of steel sheets containing cracks. *J. Mech. Phys. Solids*, 8:100–104, 1960.

[5] J.C. Gálvez, Elices M., G. V. Guinea, and J. Planas. Mixed mode fracture of concrete under proportional and nonproportional loading. *International J. Fract.*, 94(3):267–284, 1998.

[6] P. Ganguly, S. A. Vavasis, and K. D. Papoulia. An algorithm for two-dimensional mesh generation based on the pinwheel tiling. *SIAM J. Scientific Computing*, 28(4):1533–1562, 2006.

[7] Rudy J. M. Geelen, Yingjie Liu, John E. Dolbow, and Antonio Rodríguez-Ferran. An optimization-based phase-field method for continuous-discontinuous crack propagation. *International Journal for Numerical Methods in Engineering*, 116(1):1–20, 2018.

[8] M. Reza Hirmand and Katerina D. Papoulia. A continuation method for rigid-cohesive fracture in a discontinuous Galerkin finite element setting. *International Journal for Numerical Methods in Engineering*, 115(5):627–650, 2018.

[9] J. K. Knowles and E. Sternberg. Large deformations near a tip of an interface-crack between two Neo-Hookean sheets. *J. Elasticity*, 13:257–293, 1983.

[10] S. E. Leon, D. W. Spring, and G. H. Paulino. Reduction in mesh bias for dynamic fracture using adaptive splitting of polygonal finite elements. *Int. J. Numer. Meth. Engng*, 100:555–576, 2014.

[11] E. Lorentz. A mixed interface finite element for cohesive zone models. *Comput. Methods Appl. Mech. Engrg*, 198:302317, 2008.

[12] J. F. Molinari, G. Gazonas, R. Raghupathy, A.Rusinek, and F. Zhou. The cohesive element approach to dynamic fragmentation: The question of energy convergence. *Int. J. Numer. Meth. Engng*, 69:484–503, 2007.

[13] Yurii Nesterov and Arkadii Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, 1994.

[14] M Ortiz and A Pandolfi. Finite-deformation irreversible cohesive elements for three dimensional crack propagation analysis. *Int. J. Numer. Methods Eng.*, 44:1267–1282, 1999.

[15] K. D. Papoulia. Non-differentiable energy minimization for cohesive fracture. *Int. J. Fracture*, 204:143–158, 2017.

[16] K. D. Papoulia, C.-H. Sam, and S. A. Vavasis. Time continuity in cohesive finite element modeling. *Int. J. Numer. Methods Eng.*, 58(5):679–701, 2003.

[17] R. Radovitzky, A. Seagraves, M. Tupek, and L. Noels. A scalable 3d fracture and fragmentation algorithm based on a hybrid, discontinuous Galerkin, cohesive element method. *Computer Methods in Applied Mechanics and Engineering*, 200(1-4):326–344, 2011.

[18] J. R. Rice. The localization of plastic deformation in theoretical and applied mechanics. pages 207–220. North-Holland, 1976.

[19] D. Rittel and H. Maigre. A study of mixed-mode dynamic crack initiation in PMMA. *Mechanics Research Communications*, 23(5):475–481, 1996.

[20] M. J. Todd. A study of search directions in primal-dual interior-point methods for semidefinite programming. Available from `http://ecommons.cornell.edu`., 1999.

[21] Y. Wang and H. Waisman. From diffuse damage to sharp cohesive cracks: A coupled XFEM framework for failure analysis of quasi-brittle materials. *Computer Methods in Applied Mechanics and Engineering*, 299:57–89, 2016.

[22] H. Yamashita and H. Yabe. A primal-dual interior-point method for nonlinear optimization over second-order cones. *Optimization Methods and Software*, 24:407–426, 2009.

[23] Y. Ye. *Interior Point Algorithms: Theory and Analysis*. Wiley, 1997.