Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

# UGUCA: A spectral-boundary-integral method for modeling fracture and friction

David S. Kammer [a,*], Gabriele Albertini [a,b], Chun-Yu Ke [b]

[a] *Institute for Building Materials, ETH Zurich, Switzerland*
[b] *Civil and Environmental Engineering, Cornell University, Ithaca, NY, USA*

A R T I C L E   I N F O

*Article history:*
Received 21 February 2021
Received in revised form 3 June 2021
Accepted 27 July 2021

*Keywords:*
Boundary element method
Fracture
Friction
Decohesion

A B S T R A C T

Uguca is a C++ parallel implementation of the spectral-boundary-integral method to model the dynamic failure of interfaces between two elastic half-spaces. Due to its computational efficiency, uguca is suitable for fundamental research on dynamic fracture mechanics, decohesion of composite interfaces, and the onset of frictional sliding. Therefore, its potential applications range from engineering sciences to earthquake mechanics modeling. The code architecture of uguca enables straight-forward implementation of additional constitutive interface laws, which provides the user with the option of tailoring the interface mechanics to the physics of their interest.

## Code metadata

| | |
|---|---|
| Current code version | v0.9 |
| Permanent link to code/repository used for this code version | https://github.com/ElsevierSoftwareX/SOFTX-D-21-00041 |
| Code Ocean compute capsule | none |
| Legal Code License | LGPL v3 |
| Code versioning system used | git |
| Software code languages, tools, and services used | C++, FFTW, MPI, GSL, cmake, python |
| Compilation requirements, operating environments & dependencies | cmake, FFTW, MPI, GSL |
| If available Link to developer documentation/manual | https://uguca.gitlab.io/uguca/ |
| Support email for questions | uguca@ifb.baug.ethz.ch |

## 1. Motivation and significance

Mechanical failure of interfaces is a crucial physical phenomenon occurring in many engineering problems and natural systems ranging from nanotechnology up to geophysics. Interfacial failure may occur when materials break and solids slide against each other; and it does so in various forms such as fracture, decohesion and friction. While failure has been studied for decades, our fundamental understanding remains incomplete. However, studying fracture, decohesion and friction experimentally is generally challenging because of their unstable and dynamic nature and their highly localized character. Numerical simulations are needed to overcome these challenges and to provide the opportunity of gaining crucial insight to the underlying physical mechanisms governing the emergence and evolution of localized failure.

Simulation of interface failure based on boundary-element methods are common due to their computational efficiency, and various approaches have been developed over the years [1,2]. The spectral-boundary-integral (SBI) method [3,4], which is implemented in *uguca*, is particularly suitable for solving the elastodynamic equations between two elastic half-spaces with high precision. Therefore, the SBI method is an excellent tool to model and study systematically the dynamic propagation of a crack as well as the onset of frictional sliding with high resolution. It hence enables exploration of large ranges of parameter space without sacrificing numerical precision and tackle spatially and temporally multi-scale problems.

While the spectral-boundary-integral method is limited to problems with a flat fracture plane, it can be applied to model

* Corresponding author.
*E-mail address:* dkammer@ethz.ch (David S. Kammer).

problems that are complex in terms of stress distribution, material contrast and interface heterogeneities across various length scales. For instance, the SBI method has been used to simulate various failure mechanisms, including bi-material fracture [5], and dynamic cohesive fracture in heterogeneous interfaces [6]. The SBI method has also been adapted for earthquake cycle simulations [7], which are known to be extremely computationally demanding and nearly impossible to be achieved with more traditional methods, such as the finite-difference or finite-element methods. Applicability and importance of the SBI method has also been confirmed through a community code verification exercise [8].

The implementation of the SBI method presented in this paper is named *uguca* and has been applied in recent work. For instance, *uguca* was used to simulate the propagation of frictional rupture fronts, as observed in laboratory experiments, which enabled the discovery of the super-shear equation of motion [9], the limits of bi-material propagation [10,11], and the stochastic nature of static friction [12,13]. Further, *uguca* was instrumental in the process of uncovering the existence of the earthquake arrest zone [14]. While *uguca* has been developed as a research code for fundamental research, it has not been released so far. Here, we present its first release.

This paper is organized as follows. The architecture and functionalities of *uguca* are presented in Section 2. In Section 3, we present a few illustrative examples demonstrating various features of *uguca* including a 2D example of dynamic fracture propagation and a benchmark problem of rate- and stat-dependent frictional interface in a 3D medium. Finally, we discuss the major impact of *uguca* in Section 4 and provide a conclusion in Section 5.

## 2. Software description

The *uguca* code is a C++ implementation of the spectral-boundary-integral method [3,4] for the fully dynamic simulation of failure along an interface between two elastic half-spaces. We applied the independent formulation [15], which solves both half-spaces independently before they are coupled along the interface with a constitutive fracture/friction law. This approach enables a software architecture (see Section 2.1) that provides flexibility to the user and the developer, and supports straight-forward integration of new features.

Here, we first shortly summarize the underlying method. Following [4] the elastodynamic response of a 3D elastic half space is given by

$$\tau_i(x_1, x_3, t) = \tau_i^0 - \eta_{ij}^{\pm}\dot{u}_j^{\pm}(x_1, x_3, t) + f_i^{\pm}(x_1, x_3, t), \qquad (1)$$

where $x_i$ is a Cartesian coordinate system, $\tau_i$ is the traction at the surface of the half space, which lies in the $x_1, x_3$ plane, $\tau_i^0$ is the far field traction, $\dot{u}_i$ is the particle velocity, $\eta_{ij}$ the "radiation damping" coefficient matrix, and $f_i$ is a spatiotemporal integral term. $f_i$ is computed in Fourier space, using

$$f_i(x_1, x_3, t) = F_i(t; k, m)\mathrm{e}^{i(kx_1+mx_3)}\ , \qquad (2)$$

and

$$u_i(x_1, x_3, t) = U_i(t; k, m)\mathrm{e}^{i(kx_1+mx_3)}\ , \qquad (3)$$

where $\mathrm{q} = (k, m)$ is a two-dimensional wave vector, and $F_i$ and $U_i$ are the Fourier coefficients. $F_i(t; k, m)$ are computed with a series of convolution integral over the past displacement history $U_i(t - t'; k, m)$, where $t'$ are past times, with convolution kernels $H_{11}(T)$, $H_{22}(T)$, $H_{12}(T)$, and $H_{33}(T)$. The formulation for the convolution kernels can be found in [4]. We note that all three dimensions are coupled, *e.g.* $F_1(t; k, m)$ depends on $U_1(t-t'; k, m)$, $U_2(t-t'; k, m)$, and $U_3(t - t'; k, m)$.

In a nutshell, the algorithm consists in the following steps. (i) The `HalfSpace` class first computes its displacement as function of the previous displacement and velocity

$$u_i(t + \Delta t) = u_i(t) + \Delta t\, \dot{u}_i(t). \qquad (4)$$

(ii) It then computes $f_i$ in the Fourier space, as described above.

$$\text{evaluate } f_i(t + \Delta t). \qquad (5)$$

(iii) First, `Interface` class computes the interface continuity-traction $\tau_i(t+\Delta t)$ such that the displacement is continuous across the interface (*i.e.*, no crack). Then, the `InterfaceLaw` imposes a fracture criterion based on a cohesive law to decrease this continuity-traction to the strength, $\mathcal{T}$, of the interface, if it is exceeded:

$$\text{evaluate } \tau_i(t + \Delta t) = \min\left(\tau_i(t + \Delta t), \mathcal{T}\right). \qquad (6)$$

If the continuity-traction was larger than the strength, and was reduced, this will lead to crack opening and hence overall crack growth. (iv) Finally, each `HalfSpace` computes its velocity, $\dot{u}_i$, by solving Eq. (1)

$$\dot{u}_i(t + \Delta t) = \eta_{ij}^{-1}\left(-\tau_j + \tau_j^0 + f_j\right). \qquad (7)$$

The the next time step is computed by repeating this process. For increased stability, steps (iii) and (iv) can be iterated.

### 2.1. Software architecture

*uguca* is implemented as an object-oriented code. The core classes and member variables are illustrated in Fig. 1. The central class is the `Interface`, highlighted by a red frame. It contains references to objects of type (1) `Mesh` that contains information on the discretization, (2) `HalfSpace` that describes the elastic half-spaces, and (3) `InterfaceLaw` that is the implementation of the constitutive interface law. The `Mesh`, `HalfSpace`, `Interface`, and `InterfaceLaw` objects each contain nodal fields, *e.g.*, coordinates `coords`, displacements `disp`, and cohesions `cohesion`, which provide values for each discretization point.

The `Interface` class is an abstract class. Specific implementations are created by inheritance from it. The most general implementation is the `BimatInterface`, which describes an interface between two half-spaces of possibly different material properties. Other interfaces take advantage of symmetries and properties of a given problem to decrease computational cost by 50% by modeling only one half space. For example, the `DefRigInterface` assumes that the bottom material is orders of magnitude stiffer than the top material, hence it is assumed to be rigid and its displacement remains zero (it is not computed). These assumptions need to be taken into account when implementing the methods of the specific interface.

Further, the `InterfaceLaw` class, which is the most important class for users and developers alike, is also abstract. Each specific constitutive interface law is implemented as a class that inherits from `InterfaceLaw`. The specific classes describe the constitutive relation for the strength of the interface. Four different constitutive interface laws are currently available, but implementation of further laws are straight-forward.

### 2.2. Software functionalities

The key functionalities of *uguca* include (1) the two- and three-dimensional simulation of dynamic fracture of a weak-interface between two elastic half-spaces, (2) shared-memory parallelism, (3) distributed memory parallelism, and (4) straight-forward extensibility for new constitutive interface laws.

*uguca* can solve 2D and 3D dynamic fracture propagation problems. The user simply specifies the dimension during the

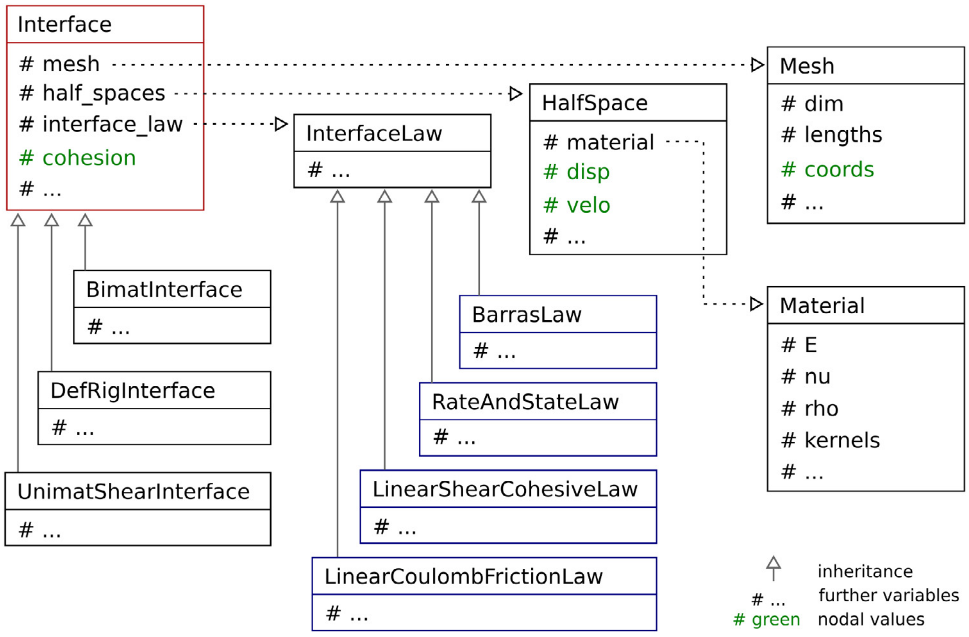

**Fig. 1.** A UML class diagram of the core of *uguca*. The constitutive interface laws, which can be easily extended by users, are highlighted by blue frames. Nodal values relevant for model set-up or analysis are marked by green font. For readability, further class variables are consolidated by `#...` and various supportive classes are omitted from this diagram.

instantiation of the class `mesh`, which was implemented by overwriting its constructor. All other methods that the user is calling directly are independent of the dimension of the problem. Hence, *uguca* is directly applicable to 2D and 3D simulations without any specific modifications needed from the user.

*uguca* is implemented with (optional) shared-memory parallelism. It applies multiple threads to compute in parallel the convolutions required for each half-space. There are 4 and 8 convolutions to be computed for each mode in 2D and 3D problems, respectively. Since, the computation of the convolutions is computationally the most demanding process in the spectral-boundary-integral method, parallelization of the convolution leads to a direct speed-up of the same order. However, it is also limited by the number of convolutions, *i.e.* 8 for 3D, and no further speed-up is possible through this approach. Nevertheless, this is particularly useful when *uguca* is run on a local computer.

For running simulations on high-performance computing facilities, *uguca* draws on its implementation of distributed-memory parallelism. Here again, the computationally demanding computation of the convolution is parallelized while all other operations are performed serially (including the fast-Fourier-transform). Parallelization is achieved by distributing the Fourier modes on different processes. This is possible because the Fourier modes are independent of each other and can be treated separately. We note that the overhead of the parallel computation consists in the communication cost related to the scatter and gather operations, which are minor and have negligible impact on the speed-up.

The implementation of *uguca* enables straight-forward extension for new user-defined constitutive interface laws. Users can implement their interface law by inheriting from the `InterfaceLaw` class and overwriting one purely virtual method that computes the interface strength. The online developer's guide [16] provides a detailed description of this process.

Finally, we highlight that *uguca* is implemented for fully elasto-dynamic problems, which is the most general case. However, the class structure of the code allows for simple extension to quasi-dynamic and static problems, which will be included in future releases. Furthermore, we note that the primary class of problems solved by the SBI method are interface failures. However, the SBI method can also model fracture of an elastic solid if crack growth is straight. In this case, the applied cohesive law represents the failure of the elastic material.

## 3. Illustrative examples

The modeling capabilities of *uguca* include both dynamic fracture propagation as well as the onset of frictional sliding; and both in two and three dimensions. To demonstrate all of these aspects, we first present an illustrative two-dimensional fracture propagation example and then a three-dimensional benchmark earthquake simulation, which is essentially a friction example. More examples and benchmark simulations are provided with the code in `uguca/examples/` and `uguca/benchmarks`, respectively.

### 3.1. 2D dynamic mixed-mode fracture

The two-dimensional fracture simulation consists of an interface between two materials with different elastic properties (*i.e.*, bimaterial interface). The interface is subjected to a static load of shear stress $\tau_0$ and normal stress $\tau_1$. The strength of the interface couples the shear and normal tractions at the interface and is described by the peak strength $\tau_{\max}$ and a characteristic length $d_c$. A seed crack with $\tau_{\max} = 0$ is introduced at the center of the interface (see Fig. 2-bottom), which leads to a spontaneous nucleation of the crack. The rupture extends naturally and propagates dynamically in both directions, as shown in Fig. 2-top. The propagation behavior to the left and right is clearly different, as can be observed from the slope and color in Fig. 2-top. This is the result of the bimaterial configuration of the modeled problem.

The properties of bimaterial fracture is known to cause clearly recognizable features [5], such as instantaneous rupture speed changes as shown at ($x \approx 0.25$ m, $t \approx 0.23$ms) in Fig. 2-top. The example shown here can be run in *uguca* by executing `uguca/build/examples/fracture_2d_example.sh`. It has 2048 degrees of freedom, runs over 859 time steps, and takes 1 second as a serial job on an Apple® M1 CPU with 16 GB

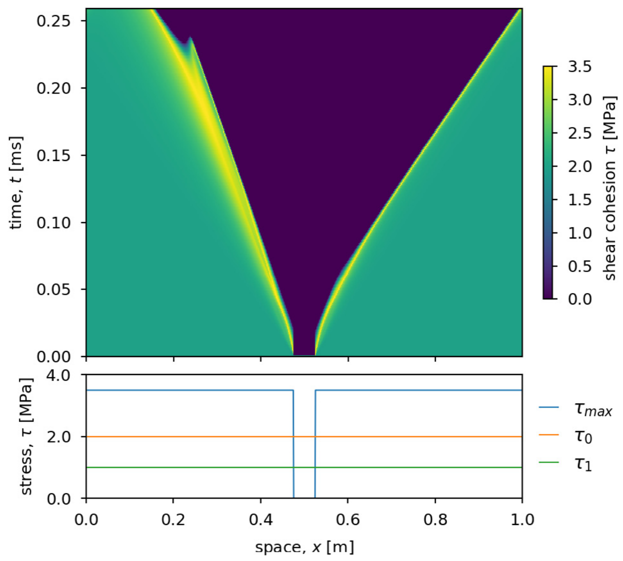

**Fig. 2.** Simulation results for 2D dynamic mixed-mode fracture propagation at a bimaterial interface. (top) Interfacial shear tractions $\tau$ (color) are shown in a space–time diagram. (bottom) Initial configuration of stresses and strength. The system is uniformly loaded by shear stress $\tau_0$ and normal stress $\tau_1$. The interface strength $\tau_{\max}$ is zero at the center ($x = 0.5 \pm 0.025$ m) and constant elsewhere. [figure created by `uguca/build/examples/fracture_2d_example_fig.py` script].

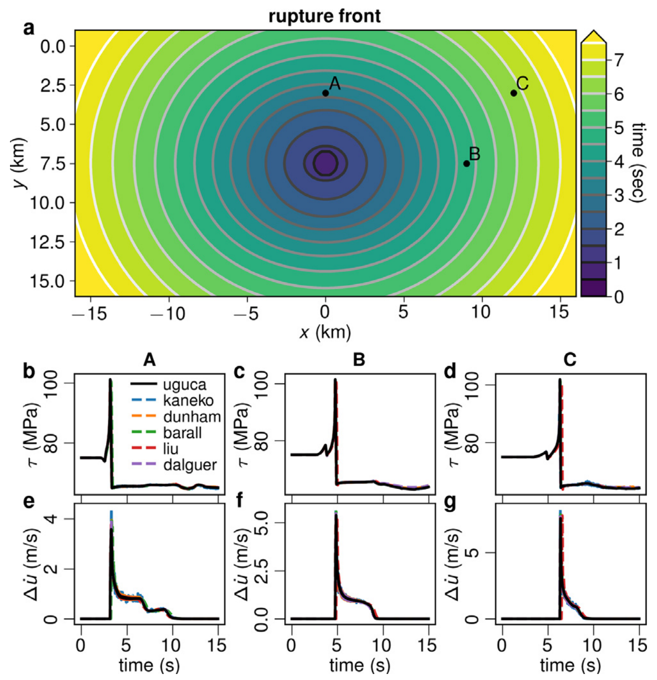

**Fig. 3.** Simulation results for 3D dynamic fracture propagation at a unimaterial interface with rate- and state-dependent friction law. (a) Position of the rupture front at different times. A, B, and C mark the locations of stations shown in (b-g) comparing the result from *uguca* to other codes. (b-d) Time history of shear stress $\tau$. (e-g) Time history of slip rate $\Delta\dot{u}$. [figure created by `uguca/build/benchmarks/TPV101/result_summary.py` script].

4266 MHz LPDDR4X SDRAM. This example illustrates the capabilities of *uguca* to model fracture problems. Comparison of *uguca* with available benchmark results is provided with the three dimensional example as presented in Section 3.2.

### 3.2. 3D earthquake benchmark simulation

For the three-dimensional friction example, we use the TPV101 benchmark problem from the SCEC/USGS Spontaneous Rupture Code Verification Project [17,18]. This benchmark is a dynamic simulation of a spontaneous rupture on a vertical strike-slip fault in a homogeneous full-space with rate- and state-dependent friction law [19,20]. A full problem description and results from other software packages are documented on the SCEC/USGS Code Verification Web Server (CVWS) at [21]. The source code for the *uguca* simulation of this benchmark problem, and other implemented benchmarks, can be found and run in `uguca/benchmarks`. This problem has 1, 036, 800 degrees of freedom, runs over 2970 time steps, and takes 1 h and 40 min with 4 cores on an Apple® M1 CPU with 16 GB 4266 MHz LPDDR4X SDRAM. Note that the solution of rate- and state-dependent friction law is implicit and hence requires Newton's iterative method for each degree of freedom at each time step, which is a computationally demanding task. We verified the implementation of *uguca* by comparing various aspects of the mechanics of earthquake ruptures with benchmark results of other codes. Our results, as shown in Fig. 3, compare well with reference solutions. Considering the time history of the slip rate (see Fig. 3e-g), we observe less numerical noise in the results of *uguca* compared to reference results from other methods, such as the finite-element and finite-difference method. This illustrates the high accuracy of the spectral-boundary-integral method, as implemented in *uguca*.

We note that the rate- and state-dependent friction law, which was empirically derived and has been widely used in the earthquake physics community, requires due to its rate dependency, an iterative solving procedure for equilibrium at each time step [22]. The implementation of the `RateAndStateLaw` showcased the versatility of *uguca*'s software architecture to accommodate complex interface laws, and the use-case of predictor–corrector time stepping methods for numerically unstable constitutive equations.

## 4. Impact

The impact of *uguca* is twofold. First, the efficiency and precision of the SBI method will continue to provide a unique opportunity to study fundamental properties of dynamic failure of interfaces. Important questions regarding the effect of local interface heterogeneities on macroscopic fracture properties, for instance, require powerful computational tools. Only boundary-based methods such as the SBI method implemented in *uguca* can provide the necessary resolution to represent the relevant mechanics and enable simulations that can bridge the full range of needed length and time scales.

Second, the most important but challenging dynamic-fracture simulations are systems with nonlinearities in the material surrounding the interface (as opposed to on the interface). These nonlinearities, which include local material inclusions, plasticity, or non-flat interfaces, are crucial for the behavior of the crack but prevent the application of the SBI method. However, other methods that can deal with material nonlinearity (*e.g.*, the finite-element (FE) method [23] and other advanced FE methods [24]) result in prohibitively large models requiring disproportionate amount of computational resources. A recently developed hybrid method [25,26], which built on *uguca*, combines the finite-element method for the nonlinear domain near the interface with the efficient SBI method for the elastic domain. This *uguca*-based hybrid method overcomes the above-mentioned limitations and presents great potential for efficient and precise simulations of earthquake source mechanics in realistic nonlinear and heterogeneous systems.

While boundary methods have been widely used in the geophysics and engineering community, there are only few implementations available as *free* and *open* software. Existing codes that are freely available online include a quasi-dynamic boundary method QDYN [27] and the fully dynamic spectral-boundary-integral method MDSBI [28]. The release of *uguca* as a free and open-source SBI code for fully dynamic simulations of fracture propagation, decohesion and frictional sliding, therefore, supports research efforts for a wide user community and strengthens the computational efforts in studying the fundamental properties of interfacial failure. Finally, adoption of *uguca* by the community is further supported by its user-friendly implementation and focus on easy plug-in capabilities for effortless implementation of user-defined constitutive interface laws.

## 5. Conclusions

We developed *uguca*, an open-source spectral-boundary-integral library, for efficient and accurate simulation of fully dynamic failure of interfaces. We verified the correctness of our implementation with multiple benchmark problems for various interface laws. The developed library supports both shared- and distributed-memory parallelisms, and is easily extensible with user-defined constitutive interface laws. Finally, *uguca* provides the possibility to be coupled with existing finite-element libraries to create a hybrid scheme, which presents great potential for modeling failure in complex media including bulk nonlinearities.

## CRediT authorship contribution statement

**David S. Kammer:** Conceptualization, Methodology, Software, Supervision, Writing – original draft. **Gabriele Albertini:** Methodology, Software, Validation, Writing – review & editing. **Chun-Yu Ke:** Methodology, Software, Validation, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgment

We are grateful to Dr. N. Richart for his help with setting up CI/CD.

## References

[1] Hong H-K, Chen J-T. Derivations of integral equations of elasticity. J Eng Mech 1988;114(6):1028–44. http://dx.doi.org/10.1061/(ASCE)0733-9399(1988)114:6(1028).

[2] Chen JT, Hong H-K. Review of dual boundary element methods with emphasis on hypersingular integrals and divergent series. Appl Mech Rev 1999;52(1):17–33. http://dx.doi.org/10.1115/1.3098922.

[3] Geubelle PH, Rice JR. A spectral method for three-dimensional elastodynamic fracture problems. J Mech Phys Solids 1995;43(11):1791–824. http://dx.doi.org/10.1016/0022-5096(95)00043-I, http://www.sciencedirect.com/science/article/pii/002250969500043I.

[4] Breitenfeld MS, Geubelle PH. Numerical analysis of dynamic debonding under 2D in-plane and 3D loading. Int J Fract 1998;93(1):13–38. http://dx.doi.org/10.1023/A:1007535703095.

[5] Barras F, Kammer DS, Geubelle PH, Molinari J-F. A study of frictional contact in dynamic fracture along bimaterial interfaces. Int J Fract 2014;189(2):149–62. http://dx.doi.org/10.1007/s10704-014-9967-z.

[6] Barras F, Geubelle PH, Molinari J-F. Interplay between process zone and material heterogeneities for dynamic cracks. Phys Rev Lett 2017;119(14):144101. http://dx.doi.org/10.1103/PhysRevLett.119.144101.

[7] Lapusta N, Liu Y. Three-dimensional boundary integral modeling of spontaneous earthquake sequences and aseismic slip. J Geophys Res: Solid Earth 2009;114(B9). http://dx.doi.org/10.1029/2008JB005934, https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2008JB005934.

[8] Erickson BA, Jiang J, Barall M, Lapusta N, Dunham EM, Harris R, et al. The community code verification exercise for simulating sequences of earthquakes and aseismic slip (SEAS). Seismol Res Lett 2020;91(2A):874–90. http://dx.doi.org/10.1785/0220190248, https://pubs.geoscienceworld.org/ssa/srl/article-abstract/91/2A/874/580609/The-Community-Code-Verification-Exercise-for.

[9] Kammer DS, Svetlizky I, Cohen G, Fineberg J. The equation of motion for supershear frictional rupture fronts. Sci Adv 2018;4(7):eaat5622. http://dx.doi.org/10.1126/sciadv.aat5622.

[10] Shlomai H, Kammer DS, Adda-Bedia M, Fineberg J. The onset of the frictional motion of dissimilar materials. Proc Natl Acad Sci 2020. http://dx.doi.org/10.1073/pnas.1916869117.

[11] Shlomai H, Kammer DS, Adda-Bedia M, Arias RE, Fineberg J. Unstable cracks trigger asymptotic rupture modes in bimaterial friction. J Mech Phys Solids 2021;149:104330. http://dx.doi.org/10.1016/j.jmps.2021.104330.

[12] Albertini G, Karrer S, Grigoriu MD, Kammer DS. Stochastic properties of static friction. J Mech Phys Solids 2021;147:104242. http://dx.doi.org/10.1016/j.jmps.2020.104242.

[13] Schär S, Albertini G, Kammer DS. Nucleation of frictional sliding by coalescence of microslip. Int J Solids Struct 2021;225:111059. http://dx.doi.org/10.1016/j.ijsolstr.2021.111059.

[14] Ke C-Y, McLaskey GC, Kammer DS. The earthquake arrest zone. Geophys J Int 2021;224(1):581–9. http://dx.doi.org/10.1093/gji/ggaa386.

[15] Geubelle PH, Breitenfeld MS. Numerical analysis of dynamic debonding under anti-plane shear loading. Int J Fract 1997;85(3):265–82. http://dx.doi.org/10.1023/A:1007498300031, https://link.springer.com/article/10.1023/A:1007498300031.

[16] Kammer D, Albertini G, Ke C-Y. uguca online user's and developer's guide, https://uguca.gitlab.io/uguca/.

[17] Harris RA, Barall M, Archuleta R, Dunham E, Aagaard B, Ampuero JP, et al. The SCEC/USGS dynamic earthquake rupture code verification exercise. Seismol Res Lett 2009;80(1):119–26. http://dx.doi.org/10.1785/gssrl.80.1.119.

[18] Harris RA, Barall M, Aagaard B, Ma S, Roten D, Olsen K, et al. A suite of exercises for verifying dynamic earthquake rupture codes. Seismol Res Lett 2018;89(3):1146–62. http://dx.doi.org/10.1785/0220170222.

[19] Dieterich JH. Modeling of rock friction: 1. Experimental results and constitutive equations. J Geophys Res 1979;84(B5):2161. http://dx.doi.org/10.1029/JB084iB05p02161.

[20] Ruina A. Slip instability and state variable friction laws. J Geophys Res: Solid Earth 1983;88(B12):10359–70. http://dx.doi.org/10.1029/JB088iB12p10359.

[21] USGS, The SCEC/USGS spontaneous rupture code verification project, https://strike.scec.org/cvws/.

[22] Lapusta N, Rice JR, Ben-Zion Y, Zheng G. Elastodynamic analysis for slow tectonic loading with spontaneous rupture episodes on faults with rate- and state-dependent friction. J Geophys Res: Solid Earth 2000;105(B10):23765–89. http://dx.doi.org/10.1029/2000JB900250, arXiv:0402594v3.

[23] Reddy JN. An introduction to the finite element method. McGraw-Hill series in mechanical engineering, 3rd ed. New York, NY: McGraw-Hill Higher Education; 2006.

[24] Nguyen-Xuan H, Liu G, Bordas S, Natarajan S, Rabczuk T. An adaptive singular ES-FEM for mechanics problems with singular field of arbitrary order. Comput Methods Appl Mech Engrg 2013;253:252–73.

[25] Ma X, Hajarolasvadi S, Albertini G, Kammer DS, Elbanna AE. A hybrid finite element-spectral boundary integral approach: Applications to dynamic rupture modeling in unbounded domains. Int J Numer Anal Methods Geomech 2019;43(1):317–38. http://dx.doi.org/10.1002/nag.2865.

[26] Albertini G, Elbanna A, Kammer DS. A three-dimensional hybrid finite element – spectral boundary integral method for modeling earthquakes in complex unbounded domains. 2021, arXiv:2102.08756 [Physics].

[27] Luo Y, Ampuero JP, Galvez P, van den Ende M, Idini B. QDYN: A Quasi-DYNamic earthquake simulator (v1.1). 2017, http://dx.doi.org/10.5281/zenodo.322459.

[28] Dunham EM. MDSBI: Multi-dimensional spectralboundary integral code (4.1.7). 2008, http://pangea.stanford.edu/~edunham/codes/mdsbi/UserGuide-v4.1.7.pdf.