dfnWorks development in support of GDSA

Spent Fuel and Waste Disposition

Prepared for
U.S. Department of Energy
Spent Fuel and Waste Science Technology
Jeffrey Hyman,
Los Alamos National Laboratory
August 2021
M4SF-21LA010304032
Los Alamos National Laboratory Report No. LA-UR-21-27931

EXECUTIVE SUMMARY

This report presents the results of work conducted between January 2021 and August 2021 at the Los Alamos National Laboratory in support of the GDSA Modeling and Integration work packages of the Spent Fuel and Waste Systems Technology Campaign for the DOE-NE's Fuel Cycle Research and Development program. This report fulfills milestone M4SF-21LA010304032. Over the past year, we have supported GDSA and DECOVALEX Task F through the continued development of dfnWorks. Specifically, we have included the following generation features to the code: Fracture locations by region within the domain, hydraulic properties by fracture family, and multiple forms to define fracture orientations. Along with these generation features, we created a visual output report of the generated network. Within code development, we improved the user input checking procedures and mesh generation. We began developing a parallel version of the particle tracking transport code within the suite and improved the performance of upscaling and meshing workflows. We have released three major releases of the code during this time. Along with these releases, we have continued to maintain our online documentation.

ACKNOWLEDGEMENTS

This work was supported by the Spent Fuel and Waste Systems Technology Campaign of the Department of Energy's Fuel Cycle Research and Development Program. Los Alamos National Laboratory is managed and operated by Triad National Security, LLC, for the National Nuclear Security Administration of U.S. Department of Energy (Contract No. 89233218CNA000001).

CONTENTS

FIGURES

TABLES

No Tables

REVISION HISTORY

No Revision History

ACRONYMS

CDF	Cumulative Density Function
CTRW	Continuous Time Random Walk
DFN	Discrete Fracture Network
DECOVALEX	Developing Coupled Models and their Validation against Expirement
dfnWorks	LANL's DFN modeling software
FRAM	Feature Rejection Algorithm for Meshing
GDSA	Geologic Disposal Safe Assessment
LaGriT	Los Alamos Gridding Toolbox
LANL	Los Alamos National Laboratory
MARFA	Migration Analysis of Radionuclides in the Far Field
PA	Performance Assessment
PFLOTRAN	A Massively Parallel Reactive Flow and Transport Model for Describing Surface and Subsurface Processes
PDF	Probability Density Function
UDFM	Upscaled Discrete Fracture Matrix

This page is intentionally left blank.

1. Introduction

This document describes the development of dfnWorks in support of the modeling and integration work package. General information and specific details about dfnWorks are provided in Hyman et al. 2015. This year, we focused on additional options to define fractures within the domain and improving the user interface. Fractures within dfnWorks are defined by family. Each family is described by the following parameters: shape (disc/rectangle), location in the domain, distribution of fracture length, orientation, and hydraulic properties. Our work this year advanced our capabilities in each of these directions. Details of this capability development are described in Section 2. For the fracture location in the domain, we included a capability to place fracture centers into the sub-regions within the domain. For hydraulic properties, we developed capabilities to define these attributes (e.g., hydraulic aperture, permeability, and transmissivity) by fracture family using four functional relationships. For fracture orientations, we included two new options to define the mean orientation of the fracture families using either trend and plunge or dip and strike. In addition to these new capabilities in fracture family generation, we performed several internal advancements to the code for an improved user interface. Details of this capability are described in Section 3. We developed a new user input checking workflow that confirms the input card for network generation is formatted correctly and self-consistent. We developed an output report PDF document that provides users with a visual summary of the generated network. Along with the new features for generation and code development, we have consistently updated and maintained the online documentation for dfnWorks, which is available at https://dfnworks.lanl.gov/. We conclude with a few general remarks in section 4 that describe additional smaller advancements.

The work described in this document was developed in specific support of GDSA PA model development (Swiler et al. 2020) and DECOVALEX Task F: Crystalline PA model development (LaForce et al. 2020)

2. Fracture Family Attributes

This section describes the new capabilities for fracture attributes.

2.1 Regions

This report shall use a standard three-dimensional Cartesian coordinate system (x, y, and z). In dfnWorks, the x-direction corresponds to east/west, the y-direction north/south, and z is vertical. The center of the domain is the point 0,0,0. Therefore values of x, y, and z are both positive and negative.

Previously, fracture families in dfnWorks had two options to specify their location in the domain. They could either be uniformly distributed throughout the entire domain or placed within vertical layers uniformly distributed therein. These vertical layers are defined using upper and lower vertical boundaries provided by the user. The upper and lower boundaries are defined using values of z-min and z-max.

This year we added an option to specify fracture family locations within cuboid regions. Each region is defined by a bounding box. The bounding box corners are defined using the eight unique triple combinations of x-min, x-max, y-min, y-max, z-min, z-max. Namely: (x-min, y-min, z-min), (x-max, y-min, z-min), (x-min, y-max, z-min), (x-max, y-max, z-min), (x-min, y-min, z-max), (x-max, y-min, z- max), (x-min, y-max, z- max), (x-max, y-max, z- max). Fractures from a family assigned to a particular region will have a center within this bounding box. Fractures can extend beyond the boundaries of the region. Similarly, fractures from elsewhere in the domain can extend into each region. There is no limit to the number of regions. Values of the regions bounding box do not need to match, i.e., the absolute value of x-min and x-max do not need to be equal. In other words, the region does not need to be a cube. The center of each region can be any point within the domain so long as the region's boundaries are within the domain boundaries. Within each region, the number of fractures can either be directly prescribed or assigned using

a fracture intensity, defined using P_{32}. If fracture intensity is used, then computation of P_{32} is performed using volume is only that of the region rather than the entire domain.

This capability to define a fracture family's centers by region facilitates several new conceptualizations in model development. Specifically, this function allows for fracture properties, such as density and intensity, to vary throughout the domain size in accordance with spatially variable geological features or regions of importance. For example, in locations within a domain of high importance, e.g., close to a well or a repository, a lower value for the minimum fracture radius can be include to more accurately honor the site characterization locally. In many cases, the domain of interest is too large to include all fractures down to the assigned minimum size. However, this region function allows one to have spatially variable fracture intensity based on these locations of high importance. Thus, the capability enhances the realism achievable using the code in a computationally feasible manner.

2.1.1 Region Example

Figure 1 shows a DFN composed of two fracture families in a domain with lengths of 20 m in all directions. Fracture family 1 is colored red, and fracture family 2 is colored blue. Family 1 is placed within a sub-region at the center of the domain with a bounding box defined by the values (-5, 5, -5, 5, -5, 5). The region boundaries are shown using the edges of the black cube in the center of the domain. Family 2 is distributed uniformly throughout the domain. Family 2 has been cut away along the xz plane for visual clarity. Notice that all fractures centers from family 1 are within the region.

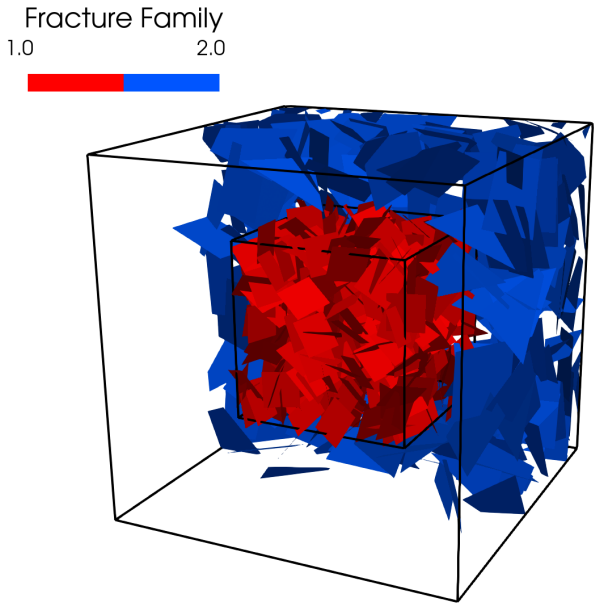

Figure 1. A DFN composed of two fracture families. Family 1 is red and is placed with a region (sub-domain) whose boundaries are depicted with the black cube in the middle of the domain. Family 2 is colored blue and is distributed throughout the whole domain. Family 2 has been cutaway along a center plane for visual clarity.

2.2 Hydraulic Properties by Fracture Family

The next capability we developed is the ability to define hydraulic properties of each fracture family according to several functional relationships that are commonly used in DFN modeling; see Hyman et al.

2016 for a discussion of the models used here and examples of their application. We developed this capability for the four functional relationships that we refer to as (1) Perfectly correlated, (2) semi-correlated, (3) uncorrelated, and (4) constant. Previously, dfnWorks users had to write their own scripts to assign these values. We streamlined these functionalities into the python wrapper, referred to as pydfnWorks. Now, users can pick any of these four functions, provide the required parameters, and the code will adjust the hydraulic properties of the corresponding fractures automatically. A major advancement is the ability to define different functional relationships by fracture family, which was not previously supported. This capability allows for depth-dependent/region-dependent parameters of these functions to be easily used. This particular capability is being implemented in DECOLAVEX and GDSA PA test cases (LaForce et al. 2020, Swiler et al. 2020).

In the following, we provide the functional forms of each of these relationships. Here, they are provided in terms of the transmissivity of the fracture. However, the code supports aperture and permeability for the functional output as well. For those functional forms, one simply swaps transmissivity with aperture or permeability, and adjusts the required parameters accordingly. Along with the functional forms, we also provide a brief description of what the use of these relationships implies about the system.

2.2.1 Perfectly Correlated Model

The first model we consider is a perfect powerlaw correlation between the fracture transmissivity T and the radius R (length) of the fracture

$$\log(T) = \log(\alpha R^{\beta}) \tag{1}$$

where α and β are parameters.

Adoption of the perfectly correlated model implicitly assumes a fairly low degree of uncertainty about the relationship between fracture size and transmissivity. While there are indications that such a relationship between size and transmissivity is useful, the deterministic formulation is an idealization. Here, all fractures of the same size are assigned the same transmissivity. Such a model disregards all mechanical, chemical, and hydrological processes that can result in variation between transmissivity of fractures of the same size. However, the formulation is convenient because each realization of network geometry requires a single realization of the transmissivity field. This functional relationship is commonly referred to as the perfectly correlated model.

2.2.2 Semi-Correlated Model

The second model we consider includes a stochastic term into the perfectly correlated model

$$\log(T) = \log\left(\alpha R^{\beta}\right) + \sigma_T \mathcal{N}(0,1) \tag{2}$$

to account for uncertainty and variability between fractures of the same size. The strength of the stochastic term is determined by the variance of a log-normal distribution σ_T and the stochastic term is an independent identically distributed random variable sampled from a normal distribution with mean 0 and variance 1, $\mathcal{N}(0,1)$. This model results in a log-normal distribution of fracture transmissivities around a positively correlated power law mean. This functional relationship is referred to as the semi-correlated model.

The semi-correlated model attempts to address some of the issues associated with its deterministic counterpart presented above. While still not resolving the physical processes that cause variations between the transmissivities of fractures of the same size, it includes a stochastic term to account for these variations. While this term might increase the realism of the network, it is more cumbersome than the perfectly correlated model, both in terms of computational demands (multiple transmissivity field realizations for the same network geometry are required) and calibration data (the semi-correlated model requires additional support from field data to constrain its additional parameters). The semi-correlated model is the least common of the models implemented in dfnWorks due to these complications, even though some researchers consider it the most realistic.

2.2.3 Uncorrelated Model

The third model assumes that there is no correlation between the fracture size and transmissivity and all values are independent identically distributed random variables from a log-normal distribution with specified mean μ_T and variance σ_T

$$\log(T) = \mu_T + \sigma_T \mathcal{N}(0,1) \tag{3}$$

This functional relationship is referred to as the uncorrelated model.

In the uncorrelated model, a lognormal distribution around a prescribed mean transmissivity is used to include variability between fractures. Here, the mechanical, chemical, and hydrological processes are assumed principally responsible for variations between fracture transmissivities rather than assuming a correlation between a fracture's size and its transmissivity. In other words, this model is the opposite extreme of the perfectly correlated model, and the semi-correlated is a combination of the two. This model is particularly useful to include hydraulic variation between fractures in networks monodisperse (uniformly sized) fractures, see Hyman and Jiménez-Martínez 2017 and Kang et al. 2020 for examples.

2.2.4 Constant Model

In the fourth model it is assumed that there is no relationship between transmissivity and radius and there is no variation between fractures. Rather the transmissivity of all fractures is assigned a single value

$$\log(T) = \mu_T \tag{4}$$

This functional relationship is referred to as the constant model.

The constant model is considered by most to be unrealistic. However, it can provide a baseline for modeling efforts to explore the impact of various other network attributes without the additional complications of varying transmissivity between fractures.

2.3 Fracture Orientation

Previously, fracture orientations in dfnWorks were only able to be defined using spherical coordinates. In the past year, we included two additional methods to define the mean orientation of a fracture family; trend/plunge and dip/strike. While spherical coordinates are convent from a modeling point of view, data from a site characterization are rarely provided in this manner. Therefore, the user previously had to convert from trend/plunge and dip/strike into spherical coordinates, which could introduce errors and was cumbersome. In this section, we provide the details of how fracture orientations can now be defined. For ease of comparison, we report the information in terms of normal vectors. We also include the information for spherical coordinates for completeness.

As part of the input checking discussed in the next section, we also included restrictions on how the values of trend/plunge and dip/strike are input. Values for those two options must be in degrees, while those for spherical coordinates could be defined using either radians or degrees. In addition to assigning stochastic fracture family orientations using trend/plunge and dip/strike, we have also added the option to defined deterministic fractures, referred to as user-defined fractures in dfnWorks, using these options. This additional option allows for more geological settings such as faults to be readily included.

2.3.1 Spherical Coordinates

Spherical coordinates are defined using two parameters ϕ and θ. ϕ is the angle that the projection of the normal vector of a fracture onto the x-y plane makes with the x-axis. θ is angle the normal vector of the fracture makes with the z-axis. From these two parameters we have the following relationship with the normal vector of the fracture $\vec{n}$

$$\vec{n}_x = \sin\theta\cos\phi \quad (5)$$

$$\vec{n}_y = \sin\theta\sin\phi \quad (6)$$

$$\vec{n}_z = \cos\theta \quad (7)$$

2.3.2 Trend and Plunge

Trend is defined as the projection of a line onto the horizontal plane and the plunge as the vertical angle measured from the horizontal downwards to this line. From these two parameters we have the following relationship with the normal vector of the fracture $\vec{n}$

$$\vec{n}_x = \cos(\text{trend})\cos(\text{plunge}) \quad (8)$$

$$\vec{n}_y = \sin(\text{trend})\cos(\text{plunge}) \quad (9)$$

$$\vec{n}_z = \sin(\text{trend}) \quad (10)$$

2.3.3 Dip and Strike

Although dip and strike are typically used to describe a single fracture's orientation, such as a fault, we provided the option for the mean orientation of stochastic fracture families as well. From these two parameters we have the following relationship with the normal vector of the fracture $\vec{n}$

$$\vec{n}_x = \sin(\text{dip})\sin(\text{strike}) \quad (11)$$

$$\vec{n}_y = -\sin(\text{dip})\cos(\text{strike}) \quad (12)$$

$$\vec{n}_z = \cos(\text{dip}) \quad (13)$$

3. User Interface

In addition to adding the previously discussed capabilities, we also improved internal aspects of the code structure to improve the user interface and interaction with the code. In this section, we describe the internal code advancements to improve the user experience with dfnWorks.

3.1 Input Checking

The input for fracture generation in dfnWorks contains many variables, which are necessary as DFN are complicated models. With such a large number of variables, it is relatively easy for a user to make typos or input parameters that are inconsistent. To address these potential issues, which could lead to network generation failure, we developed a python workflow to check for potential issues in the input card. The checking procedure is broken down into three main parts: (1) Parsing the user-provided input, (2) verification of the provided parameters, and (3) writing the parameters back to file in a clean format. The following paragraphs describe the details of each of these aspects.

Parsing the user input consists of reading in the user input file and ensuring that the formatting of each variable is correct. This parsing is performed using a python script within pydfnworks. If a variable is formatted incorrectly, e.g., an extra comma or missing bracket, then an error is printed to the screen with the name of the variable where the error occurred, details of the error, and an explanation of the required properties of the variable. Once this error is communicated to the user, then the program exits.

The user input for network generation contains 102 possible input variables. Of these 102 variables, 32 of them are mandatory. When parsing the input file, it is confirmed that all 32 mandatory variables are defined, and the values are provided. If a mandatory variable is not defined or a value not provided, then an error is

printed to the screen with the variable's name and an explanation of the required properties of the variable. Once this error is communicated to the user, then the program exits.

After the input has been parsed, then the values provided by the user are verified to ensure they are self-consistent. Verification is broken up into four main sections: (1) general domain parameters, (2) stochastically generated fractures, (3) user-defined fractures (4) FRAM information. Within each of these sub-routines, it is ensured that the parameters entered are self-consistent. For example, if a user defines a fractured family whose lengths follow a truncated power-law with upper and lower cut-offs, it is confirmed that the lower cut-off is less than the upper cut-off provided. Additionally, paths to input files defining user-defined fractures (which are not included in the input card) are checked to ensure that the paths are valid.

Once all parameters are verified, the inputs are written into a formatted output that does not contain user comments or extraneous information. This *clean* input is passed to the network generation module. While not necessary, this last step provides a pared-down version of the input card that is easily readable for both the generation module and users.

3.2 Network Generation Report

After network generation is complete, a summary of the network and domain is provided. Previously, there was a brief text summary accompanied by a few plots. We developed a comprehensive summary of the network in terms of distribution of fracture centers, the distribution of fracture orientations, the distribution of fracture sizes. These network attributes are presented for the entire network and each fracture family. Additionally, information about FRAM (Hyman et al. 2014) is provided so users can better understand how their selected parameters influenced network generation. Below we provide an example of the output report.

3.2.1 Network Generation Report Example

The fracture network used in this example is composed of four families defined in two layers within the domain. The fracture radii are sampled from truncated power-law distributions with unique exponents and upper bounds for fracture size. All fractures are circles. The fracture intensity for each family is unique and smaller in the lower layer of the domain. Figure 2 shows the network. The fractures are colored by family, and the prescription of different families into the different layers is clearly seen.

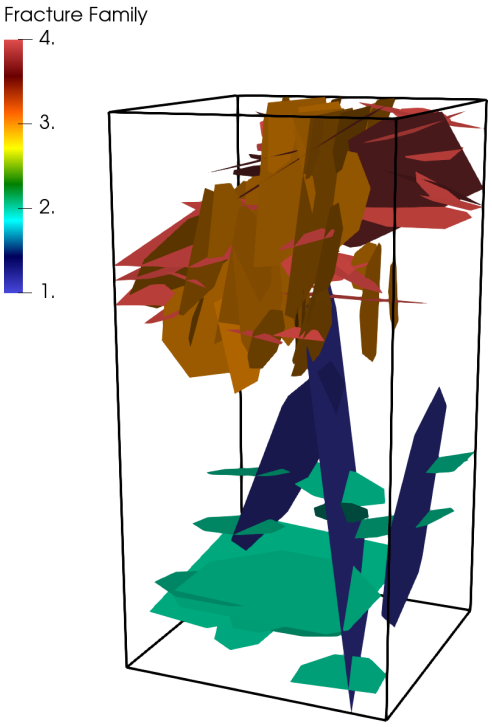

Figure 2. DFN network composed of four families in two disjoint layers. Family 1 and 2 are in a lower layer and family 3 and 4 are in an upper layer.

Figure 3 shows the first page of the report, which is a summary of the entire network. This network summary provides the essential information concerning the entire as a whole. In the upper left corner, histograms of the fracture centers are shown. The fractures are uniformly distributed in the x and y directions. The assignment of families 1 and 2 into the lower layer (negative z values) and families 3 and 4 into higher layers (positive z values) are seen in the histogram of the z component of the fracture centers. The solid black lines show the boundaries of the domain. The lower left sub-figures show the orientations of the fractures presented as the lower-hemisphere projection of their poles and a rose diagram. The upper right figure shows a histogram of the fracture radii for all families. In the lower right corner, a summary of the domain is presented that includes the number of fracture families, the total number of fractures, the final number of fractures in the connected network, the domain size, and intensity/density measurements.

In addition to the network summary, a individual report is produced for each fracture family. Figure 4 shows the summary for one family in the network (family 4). The upper left subfigure presents histograms of the fracture centers, as in the network summary. Here, there is a difference within the z-coordinate subplot. In addition to the solid lines showing the domain boundary, dashed lines indicating the boundaries of the layer to which the family is assigned are included. The lower left subfigure shows the orientations of the fracture families, which are easier to interpret than the combined version presented in the network summary due to fewer poles being shown. In the upper right subfigure, information regarding the fracture radius is presented. In the upper row, the probability density function (PDF) for both all fractures from the family (yellow) and those retained in the connected final network (blue). Along with the empirical PDFs, the analytic PDF computed using the input parameters is overlaid. The inclusion of the analytic PDF allows users to check if their target distribution shapes are being achieved. In addition to the PDFs, a single plot of the cumulative density function (CDF) is provided that includes all fractures, those retained in the final network, and the analytic value. In the lower right corner, a summary of the fracture family is provided that

includes: the number of fractures in the final network, the shape of the fractures (ellipse/rectangle), radii distribution, orientation information, and fracture intensity.

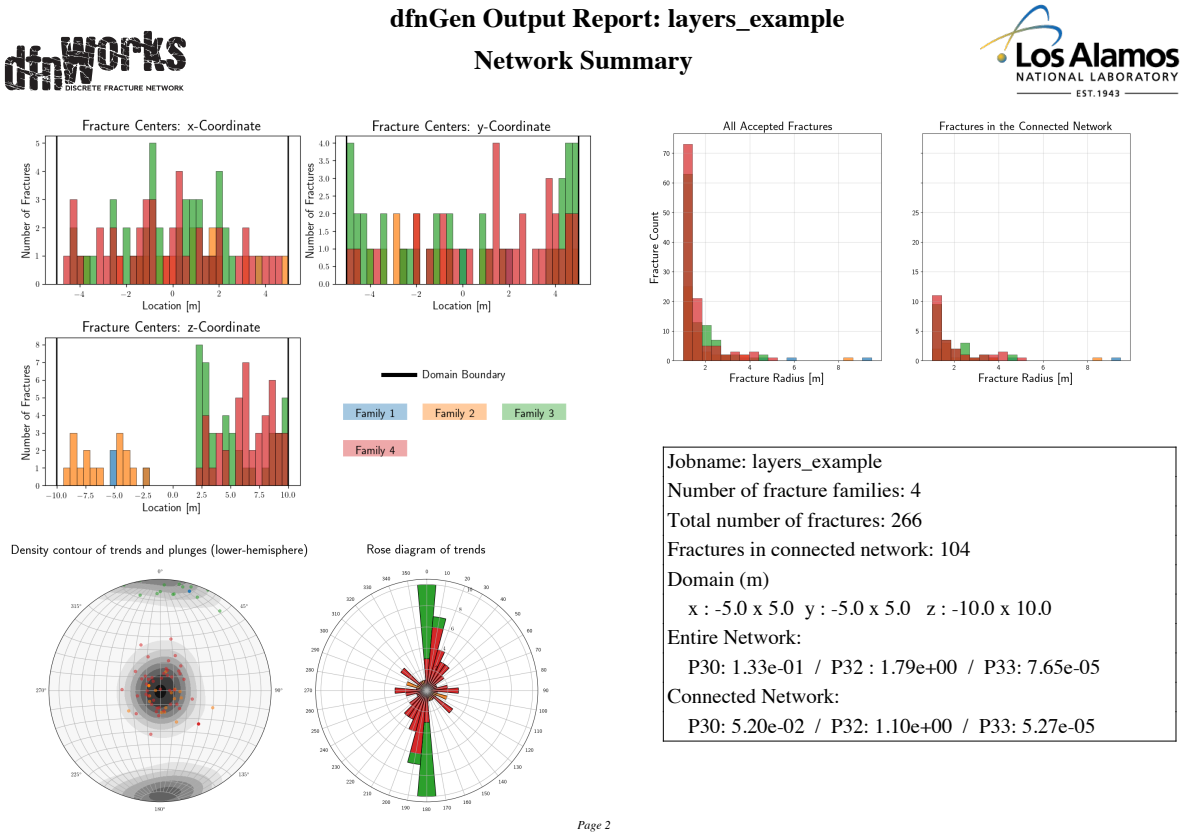

Figure 3. Combined summary of the network. In addition to the figures shown in 3.3.2-3.3.4, general information including the number of families, number of fractures, domain size, and density information is included for easy user interpretation.

The report concludes with information about how the Features Rejection Algorithm for Meshing (FRAM) influenced network generation. FRAM is a methodology that addressed the issues associated with conforming DFN mesh creation by coupling it with network generation. The cornerstone of FRAM is a user-defined minimum length scale (h) that determines what geometric features will be represented in the network. This length-scale h is conceptually equivalent to the mesh resolution. FRAM constrains the network generation so that the minimum local feature size of each fracture is greater than h. This constraint provides a firm lower bound on the required resolution of the mesh and ensures that pathological cases that degrade mesh quality are not present. In a three-dimensional DFN, measurable features include the length of the line of intersection, the distance from the end of a fracture intersection that is interior to the polygon boundary to the polygon boundary, and the distance between two fracture intersection lines segments. Once this constraint is met during network generation accomplished, all features in the network can be resolved using a mesh with edges less than h without further network modification.

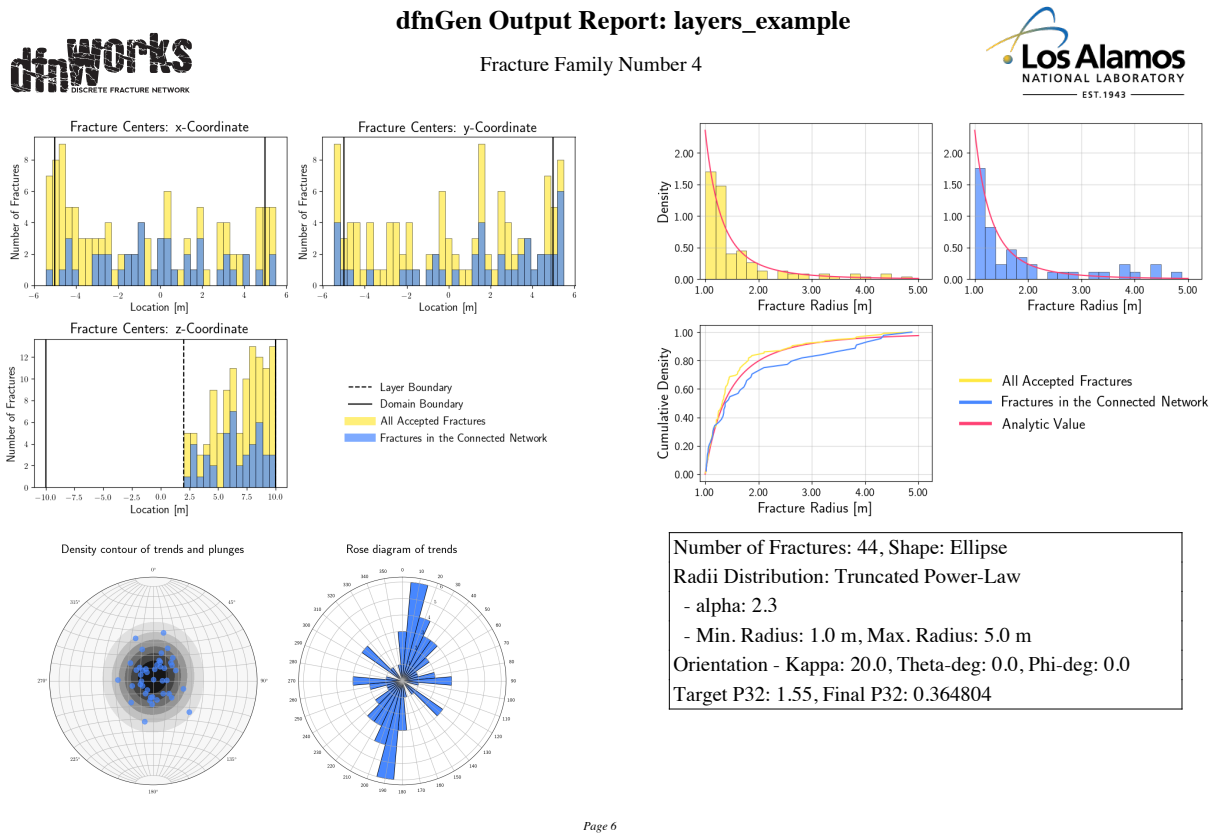

Figure 4. Information for one fracture family in the network. This is for fracture family 4 in the example.

During generation FRAM will resample and translate fractures that create features less than *h*. The bar chart shown in Figure 5 provides users with information about what were the primary causes for a fracture to be re-sampled or rejected. The most common reasons are ranked by their frequency, which is also reported. This information helps users understand how the choice of *h* has influenced network generation. Such details are critical for users to resolve potential biases that can arise during generation. The inclusion of this as a visual presentation rather than textual provides more digestable feedback.

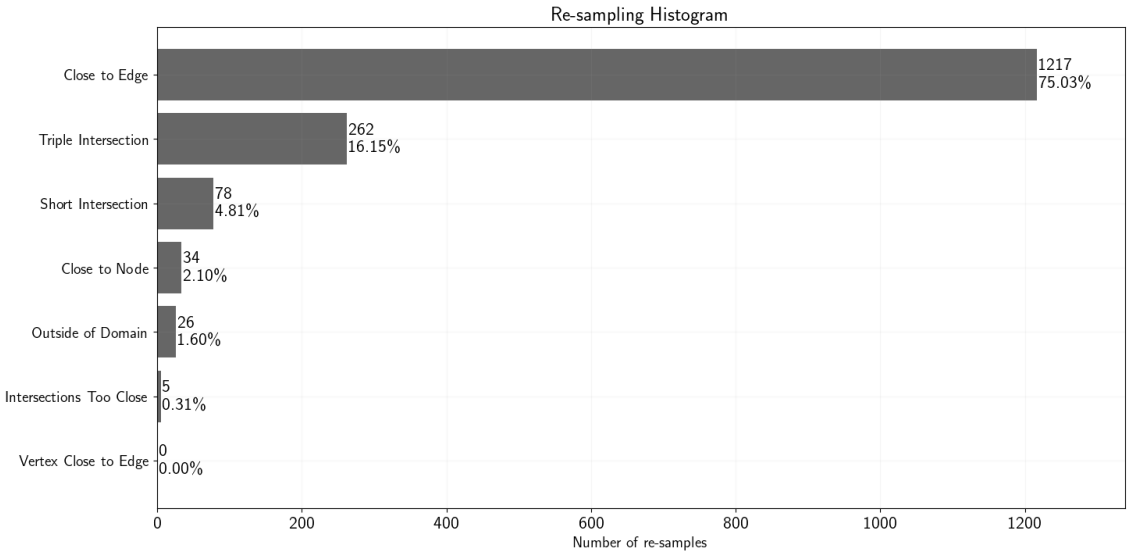

Figure 5. Generation information concerning resampling using FRAM.

4. Remarks

In addition to these aspects described in this report, we have also performed the following work. While these aspects have been essential to the success of the work package, they do not warrant lengthy discussion. In this section, we provide a few remarks about each of them for completeness.

4.1 Parallel Upscaling for the UDFM methodology

A key element of the dfnWorks software is the ability to upscale a fracture network to include both matrix and fracture properties. This methodology is referred to as the upscaled discrete fracture matrix method (UDFM); see Sweeney et al. 2019 for details. The UDFM feature is being used by DECOVALEX and GDSA for PA. Figure 6 shows one realization of a GDSA/DECOVALEX PA model with both the fracture network (colored by family and layer) and the upscaled continuum mesh (shown in grey).

In support of these projects, we updated the UDFM workflow to be more efficient in terms of lower memory and disk space usage and increased speed. To do so, we reconfigured the LaGriT meshing workflow to eliminate unnecessary i/o bottlenecks. Additionally, we rebuilt the python code that controls the LaGriT scripts' execution to be more efficient in terms of workload balance and queue organization. These changes resulted in a massive increase in speed up. Figure 7 shows the speedup performance for a simple test case consisting of four fractures, where we observed a 20% speed up, and the GDSA-PA case with 5000 fractures, where we observed a 60% speedup. These modifications were distributed in the v2.5 release of dfnWorks.

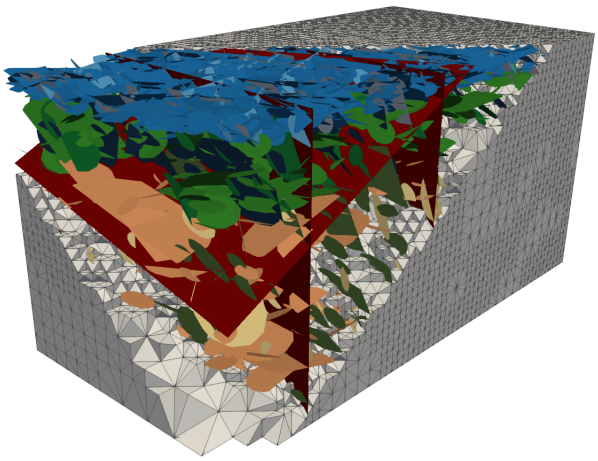

Figure 6. GDSA PA repository example with fractures and continuum mesh. The fractures are colored by family, which are assigned to three layers, and the upscaled continuum mesh is shown in grey. The UDFM method provides a variable resolution mesh that is refined close to the fractures for a more accurate representation of the network structure.

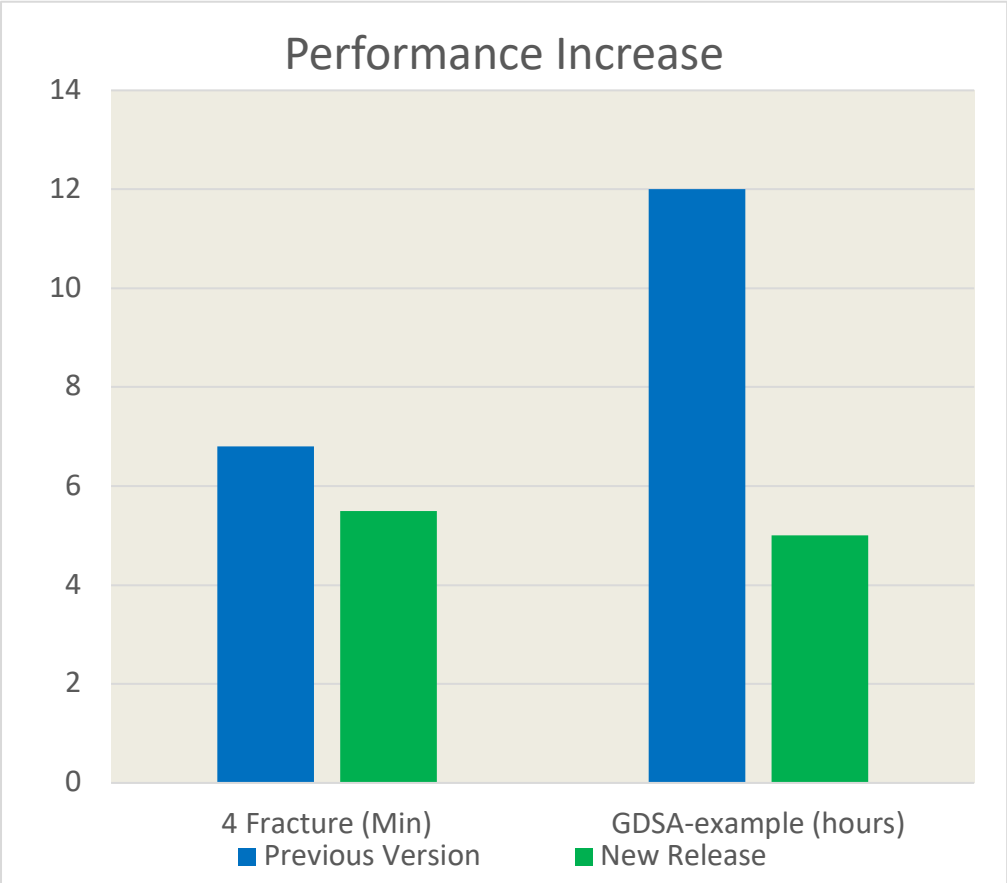

Figure 7. Code speed up due to UDFM meshing reconfiguration.

4.2 Documentation and User Support

During the development described above, we have also focused on continuous development and the upkeep of the online documentation for dfnWorks. Every time a new release is distributed, we also update online documentation at https://dfnworks.lanl.gov/intro.html. We believe that keep up to date, and accurate documentation is an essential aspect of code maintenance.

Another aspect of our work package is user support. We have interacted with dozens of users over the past year from GDSA, DECOVLAX, and other projects concerning general bug fixes and the inclusion of new features (in addition to those presented here).

4.3 dfnTrans

We have also been developing a parallel version of dfnTrans, which is the particle tracking portion of dfnWorks. Using OpenMP, we have begun preliminary development and testing. To date, we have run 250 million particles using a computer with 128 processors. Using the serial version, the largest number of particles we simulated was 1 million. This advancement is still under development but promises to be a massive improvement of the suite in the coming years.

We have also been helping support the development of a fork from dfnTrans being developed within another GDSA work package with AMPHOS21 that integrates with MARFA (Painter and Mancillas 2009) into the dfnWorks suite. The Migration Analysis of Radionuclides in the Far Field code (MARFA) uses a particle-based Monte Carlo method to simulate the transport of radionuclides in a sparsely fractured geological medium. The physical processes represented in MARFA include advection, longitudinal dispersion, Fickian diffusion into an infinite or finite rock matrix, equilibrium sorption, decay, and in-growth, which are not currently supported in dfnTrans. The advancement under development directly integrates the capabilities of MARFA as in-situ particle-based processes to be run within dfnWorks. This work is ongoing and is expected to have an open-source release within the coming year.

4.4 Transport Upscaling under Flow Heterogeneity and Matrix-Diffusion in Three-Dimensional Discrete Fracture Networks

We have also published a manuscript where we investigate the combined effects of network scale flow variability and retention due to matrix-diffusion on the scaling behavior of transport through fractured media. This study used the time-domain random walk module within dfnTrans to study two of the principal mechanisms controlling the transport of solutes through fractured low-permeability media, (1) the broad distributions of flow velocities and (2) retention times in the solid matrix. We studied the relative impact of these two processes under different initial conditions using a set of three-dimensional discrete fracture network simulations. We used these simulations to develop and calibrate an upscaled continuous-time random walk (CTRW) approach for advective transport based on an Ornstein-Uhlenbeck model for the particle velocities that accounts for the fracture-matrix coupling using a compound Poisson process. This CTRW model can be conditioned on the initial solute distribution and allows to observe late-time scaling behavior at distances beyond what is feasible to observe using high-fidelity direct numerical simulations. We determined that the initial distribution of particles leads to marked differences in the persistent long-term scale behavior in the solute travel time distributions, even those undergoing retention due to matrix diffusion through implementation and analysis of the model. This manuscript was published in Advances in Water Resources in July 2021 as Hyman and Dentz 2021.

5. References

Hyman J. D., and Dentz M. (2021) Transport upscaling under flow heterogeneity and matrix-diffusion in three-dimensional discrete fracture networks, *Advances in Water Resources*, 103994.

Hyman J. D., and Jiménez-Martínez J. (2018) Dispersion and mixing in three-dimensional discrete fracture networks: Nonlinear interplay between structural and hydraulic heterogeneity, *Water Resources Research*, 54(5), 3243-3258.

Hyman J. D., Gable C. W., Painter S. L., and Makedonska N. (2014) Conforming Delaunay triangulation of stochastically generated three dimensional discrete fracture networks: A feature rejection algorithm for meshing strategy, *SIAM Journal on Scientific Computing*, 36(4), A1871-A1894.

Hyman J. D., Aldrich G., Viswanathan H., Makedonska N., and Karra S. (2016) Fracture size and transmissivity correlations: Implications for transport simulations in sparse three-dimensional discrete fracture networks following a truncated power law distribution of fracture size, *Water Resources Research*, 52(8), 6472-6489.

Hyman J. D., Karra S., Makedonska N., Gable C. W., Painter S. L., and Viswanathan H. S. (2015) dfnWorks: A discrete fracture network framework for modeling subsurface flow and transport, *Computers & Geosciences*, 84, 10-19.

Kang P. K., Hyman J. D., Han W. S., and Dentz M. (2020) Anomalous transport in three-dimensional discrete fracture networks: Interplay between aperture heterogeneity and injection modes, *Water Resources Research*, 56(11), e2020WR027378.

LaForce, T., K. W. Chang, F. V. Perry, T. S. Lowry, E. Basurto, R. S. Jayne, D. M. Brooks, S. Jordan, E. R. Stein, R. C. Leone, and M. Nole 2020. *GDSA Repository Systems Analysis Investigations in FY2020*. SAND2020-12028R. Sandia National Laboratories, Albuquerque, NM.

Painter S., and Mancillas J. (2009) *MARFA version 3.2. 2 user's manual: Migration analysis of radionuclides in the far field*.

Sweeney M. R., Gable C. W., Karra S., Stauffer P. H., Pawar R. J., and Hyman J. D. (2020) Upscaled discrete fracture matrix model (UDFM): an octree-refined continuum representation of fractured porous media, *Computational Geosciences*, 24(1), 293-310.

Swiler, L. P., E. Basurto, D. M. Brooks, A. C. Eckert, P. E. Mariner, T. Portone, and E. R. Stein 2020. *Advances in Uncertainty and Sensitivity Analysis Methods and Applications in GDSA Framework*. SAND2020-10802R. Sandia National Laboratories, Albuquerque, NM.