SAM: Simplified Analytical Model of Dike Pathways in Three Dimensions - Instruction Manual

Lorenzo Mantiloni

GFZ German Research Centre for Geosciences, Section 2.1 ‘Physics of Earthquakes and Volcanoes’, Telegrafenberg, 14473 Potsdam, Germany.

University of Potsdam, Institute of Geosciences, Potsdam, Germany.

1 Introduction

SAM (Simplified Analytical Model) is a MatLab software performing fast and flexible simulations of three-dimensional (3D) dike pathways in a homogeneous linearly-elastic medium. The model is described in Mantiloni et al. (2023), which the user is referred to for further details.
Dikes are modeled as tensile penny-shaped cracks of fixed radius c, opening against the local direction of the least-compressive principal stress axis. Their direction of propagation is determined by finding the maximum mode I stress intensity factor (K) along the tip-line of the crack. K is proportional to the gradient of external stress and internal pressure of magma across the plane of the crack, if the external stress varies linearly in all direction (see Mantiloni et al., 2023). The model can also simulate ‘anti-buoyant’ dikes, that is, dikes with magma denser than the host rock, which tend to sink rather than ascend towards the surface. Finally, SAM can backtrack dike trajectories from a vent on the surface down through the crust. Dikes are assumed to be tensile cracks advancing through self-sustained propagation, and are not coupled to a magma chamber. The model neglects the viscous flow of magma within dikes, and does not address dike velocity or changes in dike shape. SAM needs a model of the external stress field as input.
A dike (both forward and backtracked) will stop by default if it hits the free surface, turns back or stops horizontally. Additional options may force dikes to stop under further conditions.
The software is organized as follows:

- **SAM_Main** is the main folder, where this manual is stored.
- **SAM.m** is the main code.
- **SAM_Examples** provides three scripts with examples of SAM applications. Details are provided later.
- **SAM_Functions** is a folder collecting all case-specific sub-routines. Each subroutine is associated to a different way SAM imports or computes the external stress field, either in forward

or backtrack mode. Details on each sub-routine are provided later. **SAM_Functions** includes one sub-folder:

- **Shared_functions** collects functions that are shared and frequently called in SAM sub-routines.

1.1 General Conventions

SAM simulates `N` dike pathways taking as input a model for the stress field in the host rock and `N` starting points for the dikes, the host rock and magma properties, the dike radius and, optionally, the dike volume. SAM returns the points describing the dike pathways, the orientation of the dikes (described by dip and strike angles of the dike plane) and the value of K along the tip-line of the dike at each step of the simulation. When backtracking dike trajectories, SAM returns the points where such trajectories stop in the subsurface, instead of K.
Everything is described in a Cartesian reference frame; the vertical coordinate is positive upward. '`sigma_3`' refers to the least-compressive principal stress ($\vec{v}_3$ in this Manual), while '`S3`'and '`S3dir`' stand for its magnitude ($\sigma_3 = |\vec{v}_3|$ in this Manual) and direction, respectively. All length, mass and stress/pressure quantities have, respectively, meter (m), kilogram (kg) and Pascal (Pa) as units of measure, unless specified otherwise.
Some of the modes of SAM are designed to work with the open-source Boundary-Element (BE) MatLab software 'Cut&Displace' (Davis et al., 2019), and require it to run. 'Cut&Displace' relies on the displacement discountinuity method by Crouch et al. (1983) and the analytical solutions for triangular dislocations by Nikhoo & Walter (2015), and may be found at https://doi.org/10.5281/zenodo.3694164.

Contents

2 SAM.m

2.1 Inputs

1. `Direction`: char variable setting SAM to either model a forward trajectory (`'F'`) or backtrack a dike from surface to the magma production region (`'B'`).

2. `StartPoints`: either starting points at depth (forward mode) or surface vents or arrival points of previously-simulated trajectories (backtrack mode). It must be a N x 3 array with columns being, from left to right, x,y,z coordinates.

3. `c`: radius of SAM dike (fixed).

4. `n`: number of observation points along the tip-line of the dike.

5. `Pr`: host rock density.

6. `Pm`: magma density.

7. `TopoInterp`: information about free surface and topography. For a flat surface at the datum level ($z = 0$), set `TopoInterp` to 0 or any other scalar. For non-flat topographies (or flat surfaces at $z \neq 0$), `TopoInterp` needs to be a structure with three fields, `TopoInterp.Xsurf/.Ysurf./Zsurf`. These are T x T grids of x, y and z coordinates, respectively, of points describing the topography. If the user already has an interpolating function for the free surface, it may be provided directly as `TopoInterp` (then, its class must be ‘scatteredInterpolant’ and not ‘struct’).

8. `StressModel`: structure containing the information about the model of stress field within the host rock. It must contain a `.StressOption` field. Such field must consist of one of three possible strings, associated to three modes of SAM. Each mode imports or calculates the external stress field differently, and requires different fields in `StressModel`. The modes and respective strings are:

   - `'Analytical'`: SAM relies on a user-defined function to calculate $\vec{v}_3$ (e.g. analytical expressions for stresses due to a distribution of forces on the surface of a half-space). `StressModel` must include the following fields:
     - `StressModel.Stressfun`: function handle. The user may provide any function, with one requisite: its first three arguments must be the coordinates x, y, z of the point where S3 is to be evaluated.
     - `StressModel.Stressfun_Par`: a 1 x M cell containing all the other M arguments of the function, treated as known parameters, in the order they appear in the function script.

     Example1.m illustrates this mode.
   - `'Interpolate'`: SAM relies on a grid of P observation points and stress values calculated at such points, which the user must provide. Then, SAM interpolates the stresses and use the resulting interpolating functions to run. Such functions are computed through the function **StressInterp**, which in turns relies on the MatLab function *scatteredInterpolant*. `StressModel` must include the following fields:
     - `StressModel.ObsPoints`: a P x 3 array of x, y, z coordinates of P observation points.
     - `StressModel.InterpOption`: a string which must be either `'Cartesian'` or `'S3'`. If `'Cartesian'`, SAM will interpolate the individual components of the Cartesian stress tensor S_{ij} over the grid of observation points. `StressModel` must then include the following field:
       * `StressModel.Stensor`: a P x 6 array collecting the values of stress component at the observation points: `Sxx`, `Syy`, `Szz`, `Sxy`, `Sxz`, `Syz`.

       If `'S3'`, SAM will interpolate directly $\vec{v}_3$ over the grid of observation points. The required fields for `StressModel` are then:
       * `StressModel.S3`: a P x 1 array of σ_3 values at all observation points.
       * `StressModel.S3dir`: a P x 3 array of x,y,z components of $\vec{v}_3$ at all observation points.

     Example2.m illustrates both ‘Cartesian’ and ‘S3’ modes with a pre-calculated stress model from scenario ‘Complex-Coastline’ of Mantiloni et al. (2023).
   - `'Numerical'`: SAM relies on the output of a BE numerical model for gravitational loading/unloading due to topography (Martel & Muller, 2001) to calculate $\vec{v}_3$. In particular, SAM requires the three components of displacement discontinuities on each BE (dip-slip, strike-slip, normal). This option requires ‘Cut&Displace’ to run (see Section 1.1). The synthetic scenarios of dike propagation in calderas in Mantiloni et al. (2023) were produced by employing this mode. The user must provide `StressModel` with the following fields:
     - `StressModel.MDT` sets the ‘Minimum Distance Threshold’ (MDT), a fixed vertical distance from the free surface (i.e. the nearest BE) that dikes cannot exceed, in order to prevent artifacts in the stress field. `MDT` should be equal or larger than the size of

the smallest BE in the topography mesh. Default `MDT` is set to the smallest triangular dislocation side through the function **findMDT**. This field can be optionally included in the other modes of forward SAM as well: in those cases, default `MDT` is 0.

  - `StressModel.RockPar`: elastic parameters of host rock in the following order: [Poisson's ratio (`nu`), rigidity modulus (`mu`)].
  - `StressModel.DisplMat`: a k x 3 array (k = number of BEs) whose columns contain, from left to right, dip-slip, strike-slip and normal displacements of BEs.
  - `StressModel.P1/.P2/.P3` (three separate fields): P1, P2, P3 are k x 3 arrays containing the x,y,z coordinates of the 1-st, 2-nd, 3-rd corner of each BE triangle.
  - `StressModel.TectS`: 1 x 3 array where the σ^T_{xx}, σ^T_{yy} and σ^T_{xy} components of tectonic stress are stored in this order (see the notation in Mantiloni et al., 2023).

Example3.m illustrates this mode with pre-calculated BE displacement discontinuities from scenario 'Complex-Coastline' of Mantiloni et al. (2023).

9. `options`: structure containing up to five fields. They are:

   - `options.behavior` (optional): a 1 x 3 array containing further instructions for SAM in the following order: `display=0` prevents SAM from displaying the dike trajectory (default is 1, default color of SAM cracks is red); `RigidSAM=1` prevents dikes from turning, dipping or ascending abruptly and stops them if they do (default is 0); `displast=1` makes SAM display the last iteration of the penny-shaped crack (in yellow) once the dike has already stopped (default is 0).

   - `options.cutoff`: an array with two elements: `Traj_RadialCutoff` and `Traj\_VertCutoff`. They are, respectively, the radial distance from the origin of reference frame and vertical distance from datum level where dikes are stopped. `options.cutoff` is optional in forward SAM, but must be provided in backtrack SAM or when `Pm` > `Pr` (anti-buoyant dikes, see the Introduction), otherwise dike trajectories may never stop.

   - `options.adjustdir`: scalar variable which may be optionally provided in 'Interpolate' mode. If set to 1 (default) or any other scalar $\neq 0$, it makes it so that all the user-provided $\vec{v}_3$ have positive x, y, z components. This prevents artifacts in the interpolation of $\vec{v}_3$ due to the ambiguity of sign in the components of eigenvectors. For further details about this issue, see **Appendix A - Interpolation Artifacts**.

   - `options.ProjectToFreeSurface`: an array with two elements. The first, `ToFreeSurface`, is a logical variable, while the second, `PathStep`, is a scalar. If `ToFreeSurface` is true, SAM dikes will be projected forward from the last point of the simulated trajectory until they hit the free surface. This option, set to 0 (false) by default, is helpful when working in the 'Numerical' mode, since `MDT` prevents dikes from approaching the free surface. Note that this is the case in the 'Interpolate' mode as well, if the user employs a BE model to calculate the external stress field. The trajectory projection works by fixing a direction given by the dip and strike angles of the last step in a SAM simulation, and advancing the trajectory along such a direction by a fixed increment `PathStep` (default is 50 m) until the local topographic height is exceeded.

   - `options.FromFreeSurface`: an array with two elements. The first, `FromFreeSurface`, is a logical variable, while the second, `Method`, is a scalar. If `FromFreeSurface` is true, SAM will backtrack dike trajectories which have been projected to the free surface (see the previous point about `options.ProjectToFreeSurface`). As such, it is useful when working with BE numerical models, both in 'Numerical' and 'Interpolate' mode. When this option is active, SAM provides a 'best-guess' for the point in the subsurface where the forward SAM trajectory stopped, before being projected to the free surface. This can be done in two ways, selected by setting `Method` to either 1 or 2. They are explained in detail in **Appendix B - Backward SAM from Free Surface: How to deal with the MDT.**.

10. `varargin`: up to four arguments can be assigned here. The first must be the volume `V` of the SAM dike. If `V` is a scalar, then it is assumed common for all dikes. If it is not, then it must be a N x 1 array. If SAM is not in 'Numerical' mode, the user must provide the fracture toughness (`Kc` as a variable, K_C in this Manual), the rigidity modulus (`mu`) and the Poisson's ratio (`nu`) of the host rock, in this order. When working in 'Numerical' mode, providing `Kc` is enough.

    This additional mode of SAM works only in forward, and is not described in Mantiloni et al. (2023). By assigning a volume to a SAM dike, we calculate K along its tip-line as:

    $$K = \frac{3\mu V}{4(1-\nu)c^2\sqrt{\pi c}} + \frac{4}{3\pi}\Delta\gamma c\sqrt{\pi c}, \tag{1}$$

    (cfr Mantiloni et al., 2023, Equations 5 and 6). Then, at each step of the simulation, K^{max} along its tip-line is compared to the rock fracture toughness `Kc`. If $K^{max}/K_C < 1$, the dikes will stop. This mode is included at the end of **Example1.m**.

2.2 Outputs

1. `dike_path_tip`: 1 x N cell array, each containing a M x 3 array of M points describing the N-th dike pathway. These are the x, y, z coordinates of the centers F_i, i=1,...,M, of a series of penny-shaped cracks (see Mantiloni et al., 2023).

2. `dike_path_dip`: 1 x N cell array, each containing a M x 1 array with dip angles of the dike at each point of dike trajectory.

3. `dike_path_strike`: 1 x N cell array, each containing a M x 1 array with strike angles of the dike at each point of dike trajectory. Strike angles are calculated counter-clockwise from the positive x-axis.

4. `varargout`: in forward mode, `dike_path_K` is assigned here. It is a 1 x N cell array, each cell collecting the values of the stress intensity factor `K` (see Equation 6 in Mantiloni et al., 2023) at all `n` points for all the steps of each dike trajectory. In backtrack mode, the Backtracked Starting Points (`BSP`) are assigned here. `BSP` consists of a N x 3 array of x, y, z coordinates of the backtracked starting point of each dike.

3 SAM_Functions

3.1 SAM_Forward_Analytical

This function simulates dike pathways from starting points in the subsurface to the free surface. A MDT may be optionally provided.

3.1.1 Inputs

1. `StartPoint, c, O, Pm, TopoInterp, MDT`: same as in **SAM_Forward_Interp**.
2. `Stressfun`: function handle to the analytical stress function defined by the user. Its first three input arguments must be the x, y, z coordinates of the point where we want to compute stresses.
3. `Stressfun_Par`: a 1 x M cell containing all the other M arguments of `Stressfun`, treated as known parameters, in the order they appear in the function script.
4. `options`: same as in **SAM.m** (see Section 2.1, point 9).

3.1.2 Outputs

Outputs are as described for the forward mode of SAM in Section 2.2.

3.2 SAM_Forward_Interp

This function simulates dike pathways from starting points in the subsurface to either the free surface or a MDT.

3.2.1 Inputs

1. `StartPoint`: 1 x 3 array with x, y, z coordinates of the dike starting point.
2. `c`: dike radius.
3. `O`: 3 x `n` array of observation points along the tip-line of the crack (see Figure 2b in Mantiloni et al., 2023). `O` is created within the function **SAM.m**. Rows contain, from up to bottom, x, y, z coordinates of such points, which form a horizontal ring of radius `c`, centered at the origin of the reference frame. We do this to create a 'reference' ring, which is later shifted and rotated to describe SAM cracks at specific positions and with specific orientations in space.
4. `Pm`: magma density.
5. `TopoInterp`: interpolating function (class is 'scatteredInterpolant') that gives, for a point (x,y,-z) in the subsurface, the height 'h' of topography above it, so that the vertical distance between the point and the free surface is h + z. This is useful to check whether a dike has exceeded the MDT (see next point).
6. `MDT`: Minimum (vertical) Distance Threshold from the free surface. If at least one of the `O` observation points exceeds it, the dike will stop. This is useful if the interpolated stresses come from BE numerical models (see `StressModel.MDT` in Section 2.1, point 8).
7. `options`: same as in **SAM.m** (see Section 2.1, point 9).
8. `InterpOption`: it must be one of two possible strings: `'Cartesian'` or `'S3'`. If `'Cartesian'`, the function expects interpolating functions for individual components of the Cartesian stress tensor S_{ij} as the next input arguments (see next point). If `'S3'`, the function expects interpolating functions for the magnitude and components of $\vec{v}_3$ (see next point as well).

9. varargin: if InterpOption = 'Cartesian', these must be interpolating functions for each component of the Cartesian stress tensor, in the following order: SxxInterp, SyyInterp, SzzInterp, SxyInterp, SxzInterp, SyzInterp. These functions are calculated in **SAM.m**. If InterpOption = 'S3', these must be interpolating functions for magnitude and components of $\vec{v}_3$, in the following order: S3Interp, S3xInterp, S3yInterp, S3zInterp. These functions are also calculated in **SAM.m**.

3.2.2 Outputs

Outputs are as described for the forward mode of SAM in Section 2.2.

3.3 SAM_Forward_Numerical

This function simulates dike pathways from starting points in the subsurface to a MDT (see StressModel.MDT in Section 2.1, point 8). It requires the BE software 'Cut&Displace' to run (see Section 1.1).

3.3.1 Inputs

1. StartPoint, c, O: same as in **SAM_Forward_Interp**.
2. RockPar: elastic parameters of host rock in the following order: [Poisson's ratio (nu), rigidity modulus (mu)].
3. Pr: host rock density.
4. Pm, TopoInterp, MDT: same as in **SAM_Forward_Interp**. Note that here MDT is necessary, since we are working with a BE numerical model (see StressModel.MDT in Section 2.1, point 8).
5. Dss, Dds, Dn: three k x 1 arrays (k = number of BEs) containing, respectively, the strike-slip, dip-slip and normal components of displacement for each BE.
6. P1, P2, P3: k x 3 arrays (k = number of BEs) containing the x,y,z coordinates of the 1-st, 2-nd, 3-rd corner of each BE triangle.
7. TectS: 1 x 3 array with the σ^T_{xx}, σ^T_{yy} and σ^T_{xy} components of tectonic stress, in this order (see the notation in Mantiloni et al., 2023).
8. options: same as in **SAM.m** (see Section 2.1, point 9).

3.3.2 Outputs

Outputs are as described for the forward mode of SAM in Section 2.2.

3.4 SAM_Backtrack_Analytical

This function backtracks dike trajectories from starting points on the free surface or in the subsurface down through the half-space. Backtracked trajectories stop if they turn around, become horizontal or their radial distance from the origin and/or their depth exceed given thresholds.

3.4.1 Inputs

Inputs are the same as in **SAM_Forward_Analytical**. Here, `cB` is the backtracking radius. One additional input is `Pr`, the host rock density.

3.4.2 Outputs

Outputs are as described for the backward mode of SAM in Section 2.2.

3.5 SAM_Backtrack_Interp

This function backtracks dike trajectories from starting points on the free surface or in the subsurface down through the half-space. Backtracked trajectories stop if they turn around, become horizontal or their radial distance from the origin and/or their depth exceed given thresholds.

3.5.1 Inputs

Inputs are the same as in **SAM_Forward_Interp**. Here, `cB` is the backtracking radius. One additional input is `Pr`, the host rock density.

3.5.2 Outputs

Outputs are as described for the backward mode of SAM in Section 2.2.

3.6 SAM_Backtrack_Numerical

This function backtracks dike trajectories from starting points on the free surface or in the subsurface down through the half-space. Backtracked trajectories stop if they turn around, become horizontal or their radial distance from the origin and/or their depth exceed given thresholds.

3.6.1 Inputs

Inputs are the same as in **SAM_Forward_Numerical**. Here, `cB` is the backtracking radius. One additional input is `Pr`, the host rock density.

3.6.2 Outputs

Outputs are as described for the backward mode of SAM in Section 2.2.

4 SAM_Examples

This folder contains three applications of SAM. Both forward and backtracking simulations are included.

4.1 Example1.m

This is an application of SAM in ‘Analytical’ mode. The function **HalfSpaceNormalForce**, explained later on, is the user-defined analytical function. Four dikes start from the same depth at equally-spaced points along the edge of a circular, sill-like reservoir, which does not contribute to the stress field in the host rock. Forward trajectories are then backtracked from their arrival points on the free surface (no MDT is needed here), and the average distance between actual and backtracked starting points evaluates the performance of the method. We encourage users to employ values of backtrack radius cB different than the forward radius c, and see how that impacts the performance.

4.2 Example2.m

This is an application of SAM in 'Interpolate' mode. It loads a pre-calculated stress model from the 'Complex-Coastline' scenario of Mantiloni et al. (2023). The file '**Example2_Data.mat**' is required. The host rock is stressed by the presence of a caldera lying on a coastline, with some topographic highs on the mainland. Tectonic stress is superimposed. The reservoir has no contribution to the stress field. The script first simulates one dike starting from the edge of a circular, sill-like magma reservoir, then it backtracks its trajectory. The backtrack is performed both from the point where the forward trajectory stops in the subsurface (at or close to the MDT) and from the projected arrival point on the free surface. The backtracked trajectory is stopped slightly below the reservoir depth. The two methods are eventually compared in terms of the average distance from actual and backtracked starting points of the dikes. Alternatively, the user can simulate ten dikes from imported starting points, and backtrack their trajectories as well. Finally, the forward simulation is re-run for the same starting points, but with a common dike volume assigned.

4.3 Example3.m

This application works with the same stress model of **Example1.m**, but SAM operates in 'Numerical' mode and takes pre-computed displacement discontinuities on the BEs as input. The file '**Example3_Data.mat**' is required. Running times are thus considerably longer than for **Example2.m**.

5 Shared_functions

This folder collects all sub-routines that are frequently called in **SAM_functions**. They are rather straightforward, so they are briefly mentioned here.

5.1 alpha_finder

This function retrieves the strike angle (`al`) of a crack oriented perpendicularly to $\vec{v}_3$ evaluated at its centre, knowing the dip angle (`theta`) of the crack's plane.

5.1.1 Inputs

1. `S3dir`: a 1 x 3 array with x, y, z components of $\vec{v}_3$ calculated at the center of the current SAM crack.

2. `theta`: dip angle of the current SAM crack. It must be calculated counterclockwise away from the plane z = 0 (so that for a vertical crack pointing downward it would be `theta` $= \pi/2$).

5.1.2 Outputs

1. `al`: the strike angle of the current SAM crack. It is calculated counter-clockwise away from the positive x-axis.

5.2 Retrieve_sigma3

This function calculates the magnitude and x, y, z components of $\vec{v}_3$ at given observation points. It first evaluates each component of the Cartesian stress tensor S_{ij} through 'Cut&Displace' (see Section 1.1), before retrieving its eigenvectors.

5.2.1 Inputs

1. `X, Y, Z`: coordinates of the P observation points. Each is a P x 1 array.
2. `TectS, Dds, Dss, Dn, P1, P2, P3`: same arguments as in **SAM_Forward_Numerical**.
3. `Pr, nu, mu, g`: respectively, host rock density, Poisson's ratio, rigidity modulus and acceleration due to gravity. The latter is fixed in higher-order functions such as **SAM_Forward_Numerical** and **SAM_Backward_Numerical** to 9.81 m/s^2.

5.2.2 Outputs

1. `S3, S3dir`: magnitude (P x 1 array) and direction (P x 3 array with x, y, z components) of $\vec{v}_3$ at observation points `X, Y, Z`.

5.3 Retrieve_sigma3_Interp

This function calculates the magnitude and x, y, z components of $\vec{v}_3$ at given observation points. It first evaluates each component of the Cartesian stress tensor S_{ij} through interpolating functions, before retrieving its eigenvectors.

5.3.1 Inputs

1. `X, Y, Z`: coordinates of the P observation points. Each is a P x 1 array.
2. `SxxInterp, SyyInterp, SzzInterp, SxyInterp, SxzInterp, SyzInterp`: interpolating functions for each component of the Cartesian stress tensor, same as in **SAM_Forward_Interp**.

5.3.2 Outputs

1. `S3, S3dir`: magnitude (P x 1 array) and direction (P x 3 array with x, y, z components) of $\vec{v}_3$ at observation points `X, Y, Z`.

5.4 HalfSpaceNormalForce

This function is required in **Example3.m** as an instance of user-defined analytical function to be used in 'Analytical' mode. It calculates $\vec{v}_3$ at observation points in a half-space, in the presence of a distribution of normal forces on the free surface. The distribution simulates surface loading/unloading (due to e.g. a volcanic edifice or a caldera). Solutions for the stress components are based on those for a single normal force of intensity `N` by Jaeger, Cook & Zimmerman (2007), 13.5. Inputs and outputs are explained in detail at the beginning of the script.

5.5 A few functions borrowed from 'Cut&Displace'

The functions **EigCalc3d** and **CalculateStressOnSurroundingPoints3d** are part of the 'Cut&Displace' software, and were originally developed by Tim Davis. **EigCalc3d** provides an efficient way of retrieving the principal stresses from a given stress tensor. **CalculateStressOnSurrounding-Points3d** evaluates Cartesian stress components at given observation points, taking in the displacements of the BEs and the BE mesh itself as inputs, among others. The original version included progress bars showing the advancement of calculations while it is running. Here, they are commented in order to prevent unnecessary visual output and reduce running times.
Both functions may be found in the original 'Cut&Displace' software, but we recommend employing the ones provided here for ease of use.

6 Appendix A - Interpolation Artifacts

The following example may help understand the meaning of **options.adjustdir** (see **SAM.m - Inputs - options**): let the stress eigenvector $\vec{v}_3$ calculated at points a and b be $\vec{v}_3^a$ and $\vec{v}_3^b$, both parallel to the x-axis, but pointing to opposite directions. As far as SAM is concerned, the only thing that matters is the orientation of $\vec{v}_3^a$ and $\vec{v}_3^b$, not the direction they point to. If, however, we interpolate the vector field between a and b and then calculate a vector $\vec{v}_3^c$ at the mid-point between them, it will have a null x-component. This will lead to a wrong orientation of SAM dikes, if they pass through that point. If both $\vec{v}_3^a$ and $\vec{v}_3^b$ point to the same direction (i.e. if both their x-components are positive), we can dodge this issue.

7 Appendix B - Backward SAM from Free Surface: How to deal with the MDT.

SAM dikes are stopped at a Minimum Distance Threshold (MDT) from the free surface when the topography is modelled by a mesh of Boundary Elements (BEs). Not doing so would result in dike trajectories being influenced by artifacts in the stress field due to the proximity of the BEs. This also implies that the MDT should be larger for larger sizes of the BEs. The MDT is a required input in the 'Numerical' mode and when the 'Interpolated' mode relies on a stress field produced by a BE model. When `MDT` $\neq 0$, the arrival point of a SAM trajectory will be in the subsurface. The problem is, what happens when we want to backtrack dike trajectories from points on the free surface (e.g. eruptive vents), but we rely on a BE numerical model to deal with the topography? In such a case, we must account for the MDT and find an "effective" starting point for the backtracked trajectory in the subsurface. We employ two different methods to do so. The user can switch between them by assigning 1 or 2 to the second element of the field `options.FromFreeSurface` (see Section 2.1). Both methods are implemented in the function **FirstBacktrackPointFinder**, defined in both **SAM_Backtrack_Numerical** and **SAM_Backtrack_Interp**. We describe them in the following.

1. We take the point on the free surface and shift it vertically by -MDT. This identifies "point b". Then, we calculate $\vec{v}_3$ at point b and project point b along a direction that is both perpendicular to $\vec{v}_3$ and represents the shortest path to the free surface. Once the projected point b reaches the free surface, we calculate the distance vector between it and the actual starting point (`modSPdiff`), and shift point b by that vector. We repeat the procedure for a certain number of times (default is 10) or until a specific precision (`diffthr`) is attained. We apply the method for different multiples of MDT and obtain a set of points b. Finally, we take

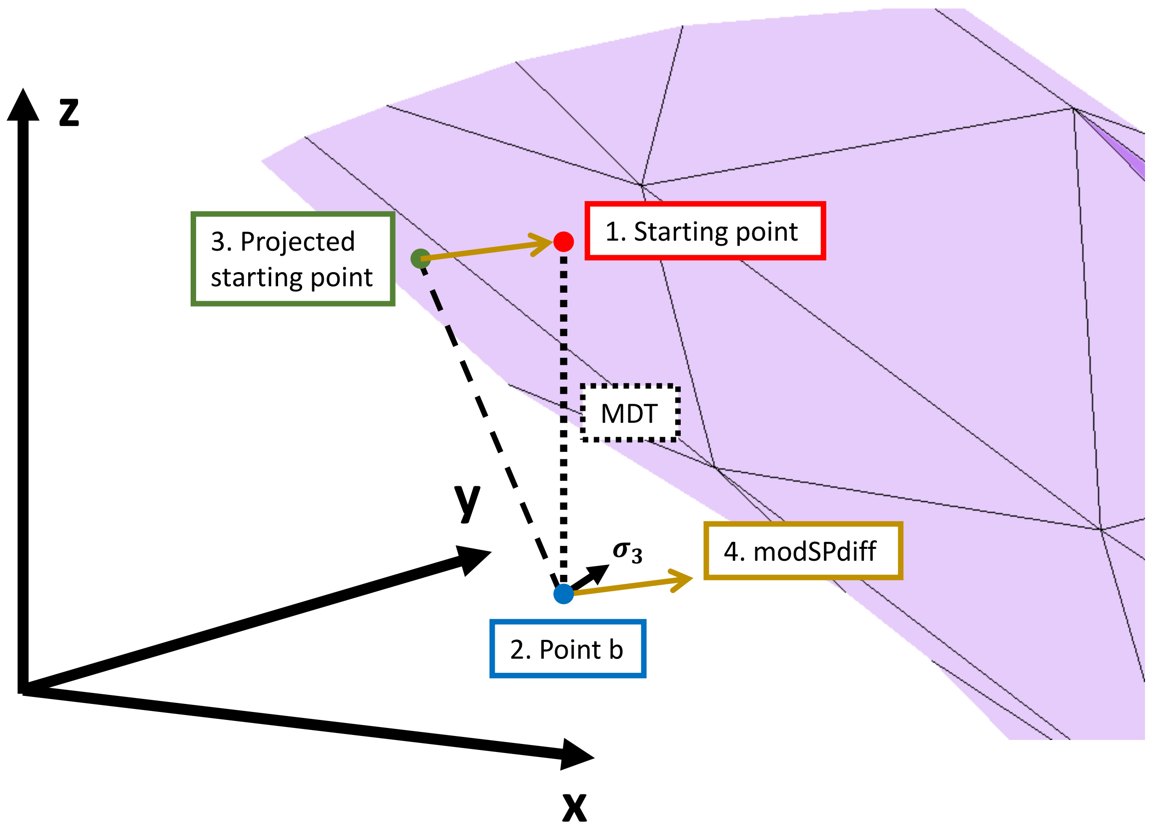

Fig. 1: Method 1 workflow. The Boundary Elements modeling the free surface are lilac triangular dislocations.

the point b associated with the smallest `modSPdiff` as the starting point and backtrack the trajectory from there. See Figure 2 for a diagram of the method.

2. We take the point on the free surface and apply a modified version of the general backtracking method of SAM. As explained in Mantiloni et al. (2023), SAM backtracks a trajectory by minimizing at each step the scalar product between $\vec{v}_3$ at a candidate point (the one we still do not know) and the vector pointing from the previous point (the one we retrieved in the previous step) to the candidate one. The distance between previous and candidate point (`cB`) is fixed, and the MatLab function *fminsearch* samples the parameter space of the dip and strike angles (`theta`,`al`). Here, we let the backtrack radius `cB` vary as well, so that *fminsearch* samples the (`cB`,`theta`,`al`) parameter space. We initialize *fminsearch* to (`k*MDT`,$\pi/2$,`al_0`), where `al_0` is the strike angle of the starting point on the free surface, and repeat the procedure for different multiples of `MDT` (`k*MDT`), retrieving a set of candidate "points b". Then, we apply the same steps as in Method 1, and obtain a "best candidate" starting point for the backtracked trajectory.

8 Acknowledgments

I am grateful to Eleonora Rivalta for the thorough supervision, feedback and support, and thank Tim Davis, Mehdi Nikkhoo and Greta Bellagamba for useful comments and debugging.

9 Software availability

The open-source Boundary-Element tool ‘Cut&Displace’ is found at
https://doi.org/10.5281/zenodo.3694164.

10 Found a bug?

I appreciate any feedback or bug report. Users can reach out to me at **lorenzo@gfz-potsdam.de**.

11 References

Crouch, S. L., Starfield, A. M., & Rizzo, F. J. (1983). Boundary element methods in solid mechanics.

Davis, T., Healy, D., & Rivalta, E. (2019). Slip on wavy frictional faults: Is the 3rd dimension a sticking point?. Journal of Structural Geology, 119, 33-49.

Jaeger, J. C., Cook, N. G., & Zimmerman, R. (2007). Fundamentals of Rock Mechanics.

Mantiloni, L., Rivalta, E., & Davis, T. (2023). Mechanical modeling of pre-eruptive magma propagation scenarios at calderas. Journal of Geophysical Research - Solid Earth.

Nikkhoo, M., & Walter, T. R. (2015). Triangular dislocation: an analytical, artefact-free solution. Geophysical Journal International, 201(2), 1119-1141.