Report

Conversion SamosAT DFA-Kernel - AvaFrame

Investigating and comparing differences between com1DFAOrig (SamosAT based) and com1DFA

funded by

Wildbach- und
Lawinenverbauung
Forsttechnischer Dienst

in cooperation with

April 2022

Authors: AvaFrame Core Team, BFW/WLV

Revision	Date	Author(s)	Description
0.1	29.1.2022	FSO, AW, MTo	1st draft
0.2	31.1.2022	AW, MTo	Feedback round
0.3	2.2.2022	JT	Feedback round; restructure
0.4	2.2.2022	FSO, JT, AW, MTo	Layout, minor changes
1.0rc1	7.2.2022	FSO, JT, AW, MTo	Typos
1.0rc2	7.3.2022	CT,MG	Corrections, Clarifaction
1.0	6.4.2022		Final version

Contents

1 Scope and resources

This report contains following items

- The current document with the report itself
- report pdf/html for *null* simulations
- report pdf/html for *entrainment* simulations
- report pdf/html for *resistance* simulations

In addition we refer to the AvaFrame documentation at `https://docs.avaframe.org`, especially for the description of all test cases. Furthermore we also point to `https://github.com/avaframe` for the source code and issue tracking.

Note: we often include (big) figures side by side for overall readability. Please use your pdf readers zoom function to get more details!

2 Summary

This document shows the comparisons and developments during the conversion from the c++ DFA SamosAT kernel, called **com1DFAOrig** (com1DFA exectuable on gitea) to the AvaFrame **com1DFA** python/cython implementation. In a perfect world the results between these two implementations would be exactly the same down to the computational precision. However in real world applications a complete match is impossible, the reasons ranging from different programming languages, slightly different programming of sub-functions and many more. The aim is to show where and how big the differences are between these two modules. We show examples for aspects we explored during the conversion. This includes different categories of sources of uncertainties, be it numerical methods, implementation methods (i.e programming) or input issues. We show that we deem the differences small enough and within acceptable ranges, however we also want to highlight the issues that arise when looking closer and explain their source.

For the attached report following versions are used:

- com1DFA: github commit 32408c5fc74dd8b (26. Jan. 2022)
- com1DFAOrig: git.avaframe.org commit c771be02e3c771b (source code; executable available on github repository com1DFA_Exe)

Please note that com1DFAOrig, the extracted DFA kernel from SamosAT, is used and not SamosAT itself. There is a difference in results already between com1DFAOrig and SamosAT (see also github issue number 109). This is suspected to be a compiler issue, the root cause is not resolved. However, while these differences are noticeable, they are small compared to the next issues.

There are two caveats regarding the direct comparison between com1DFAOrig and com1DFA:

1. Our investigations showed the initial particle distribution playing a significant role for the simulation results. Even something as simple as changing the random number seed used in com1DFAOrig leads to very noticeable differences in the runout behavior. To avoid any differences stemming from different random number generators in the different programming languages c++ and python, we decided to use following procedure for comparison:
    - modify com1DFAOrig to output its initial particle distribution to file

- run com1DFAOrig to get results and the initial particle distribution
- read and use this distribution in com1DFA, therefore *completely* bypassing the com1DFA python initial particle distribution implementation and random number generator. However this allows for a better comparison of the numerics itself.

As a sidenote: the problem with the dependency on the random number seed also exists in com1DFAOrig (and SamosAT). We already investigated alternative approaches, but so far only incremental improvements were found, not really solving the problem.

2. A second issue, similar to the previous one, is the handling of input digital elevation models (DEM). More specifically the handling of the cellsize information from the ASCII header. If the cellsize is set to something different than 5m used in both modules numerics, remeshing of the input DEM takes place. Due to slightly different implementations, this leads to ever so slightly different remeshed DEMs. However, this small change then subsequently influences the placement of particles and leads to the same problem as in the previous point. We have some test cases where we specifically use a cellsize different from 5 meters (4.9XX), with the express intent to be able to test the remeshing behaviour. Again, to allow for better comparison of the numerics itself, we reset all the DEMs to have a 5m resolution for this report to avoid remeshing.

Following we highlight the main differences on selected examples. Please note: we only show one plot per avalanche/example, please refer to the attached reports to get all figures for each case.

Waves pattern / Borders

Noticable differences show up in areas, where wavy patterns can be observed. One example is shown in figure 1, displaying peak flow depth for the helix channel test case. The area showing waves also shows the biggest (noticable) differences, however both modules produce the same patterns. This is due to a slight shift of one or two gridcells leading to more pronounced differences since the gradients are high. The same is observed at the outline of the results, where a shift of one or two gridcells again leads to differences. Effects of the same type can also be seen in *avaHit* and *avaHockeySmall*.

Steep sections

The middle part of *avaGar* with the release area *relGar2* shows a noticeable difference (figure 2). This is a part in which the topography has a very steep section, i.e. a cliff, leading to high curvatures. Higher curvatures lead to more pronounced effects of the uncertainties in curvature terms. Another issue is related to reprojecting particles onto the topography. At each time step particles need to be adjusted to the topography, in simple terms: brought back down (seldom up) to the surface (opposed to flying out in the air). There are different ways of doing this (gravity direction, perpendicular to topography, . . .). We did not match the methods between com1DFAOrig and com1DFA, so we see larger differences between the two methods in steeper areas. The differences seem to be bigger when using coulomb friction, which results in higher velocities. This means, if the time step is big, the reprojection term is also significant.

However, even with the differences in the middle of the avalanche track, the runout still is nearly identical!

Small sidearms

Figure 3 shows a sidearm developing at the top for com1DFAOrig. This occurs in very few simulations, but always with the same characteristics: it is a few particles crossing / falling down the 'wrong' side of a ridge. As this sometimes also occurs in com1DFA and is usually solved by a small adjustment in the release areas, we attribute this to small differences in particle locations and not a fundamental underlying problem. And again, since there are no differences in the runout area, we deem this acceptable.

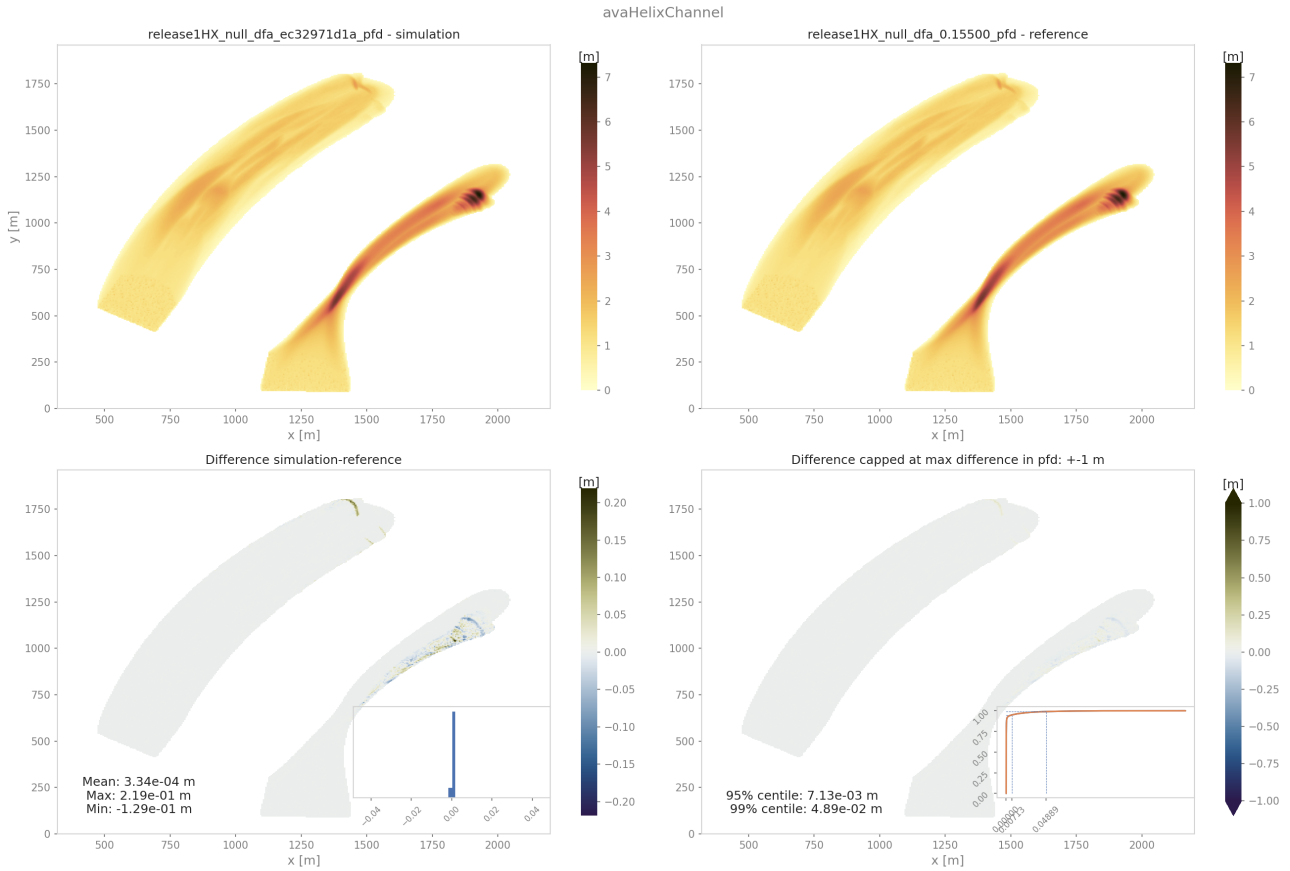

Figure 1: Peak flow depth for release 1 of the helix channel test case (idealized topography). There are 2 avalanche tracks, one with and one without channel.

2.1 Conclusion and recommendation

Looking at the attached reports for the *null*, *entrainment* and *resistance* simulations in combination with the sections in this report, it is shown that the differences in results are often not discernable by eye when looking at the full simulations. Both resistance and entrainment show differences in the order of 0.5 percent for total mass (2.5 percent for entrained mass) with some differences in the middle of the avalanche path, but with no effects on the runout area. The *null* simulation report, including all real avalanche test cases, also shows visually identical results for all peak variables.

So, despite the previously described caveats and differences, looking at all report examples the general patterns are close, if not identical. The differences we show are in an acceptable range and explainable, leading us to the conclusion that com1DFAOrig and com1DFA show similar enough behaviour. The differences shown are all well within the model uncertainties, see the section about initial particle distribution (section 7.3) for an example.

We recommend continuing the field trial phase with operational users using AvaFrame alongside their existing workflow. This ensures potential problems and issues show up in a setting with many additional use cases.

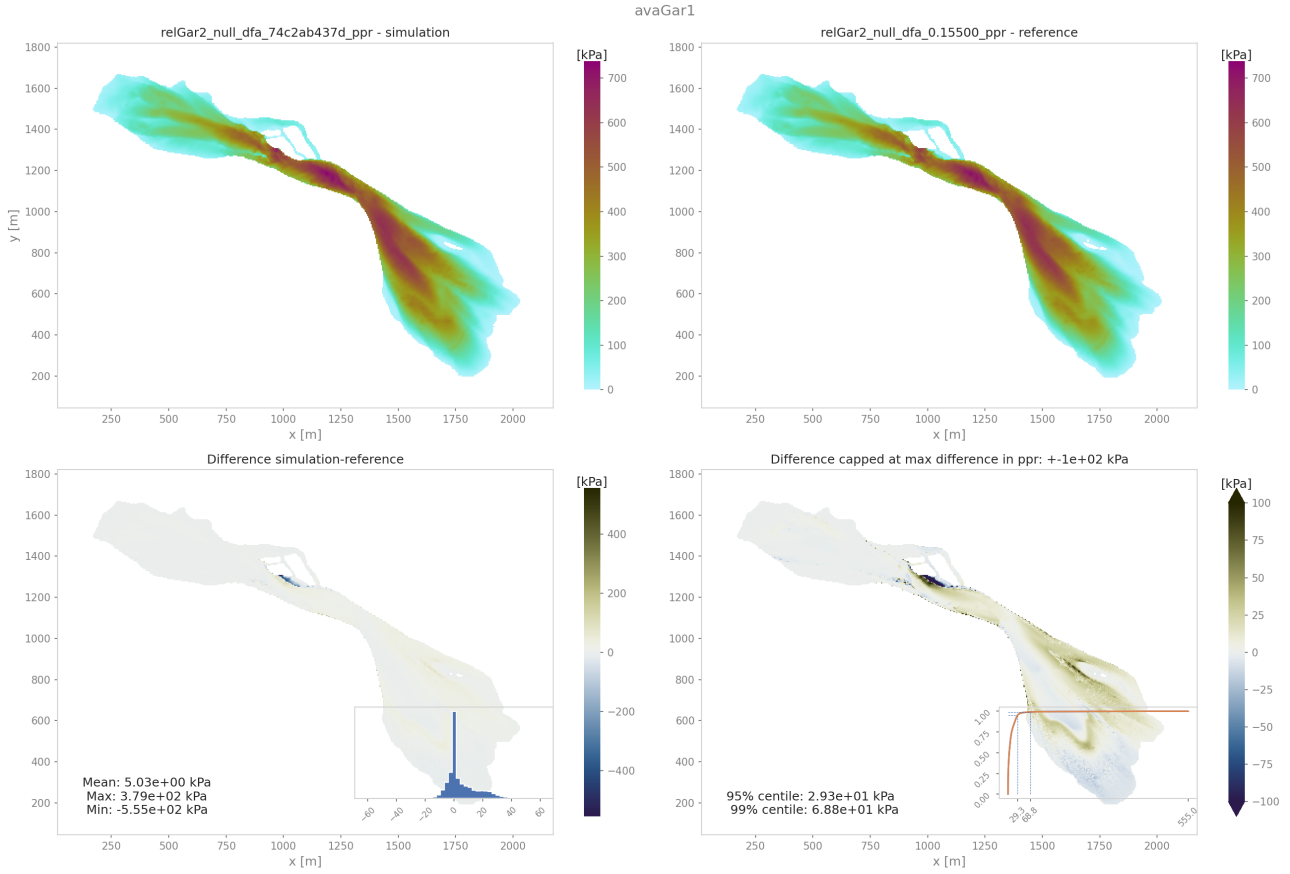

Figure 2: Peak pressure for release 2 of the avaGar test case (real topography with a very steep section one third down the track).

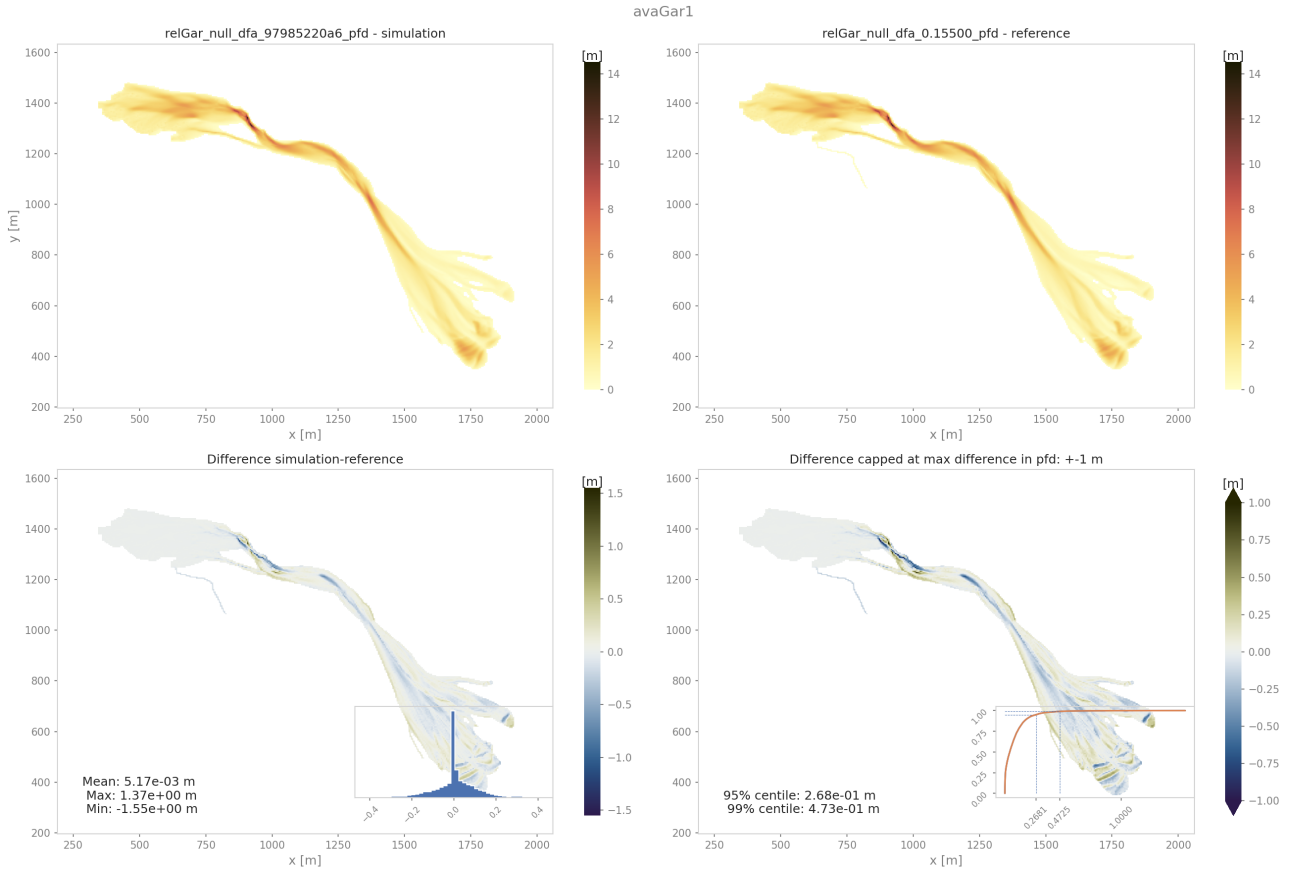

Figure 3: Peak flow depth for release 1 of the avaGar test case (real topography with a very steep section one third down the track).

3 Zusammenfassung

Das vorliegende Dokument zeigt die Vergleiche und Entwicklungen, die wir während unserer Konvertierung des c++ DFA SamosAT-Kernel, genannt com1DFAOrig (com1DFA verfügbar auf gitea), zur AvaFrame com1DFA python/cython Implementierung durchgeführt haben. In einer perfekten Welt wären die Ergebnisse genau gleich, abgesehen von der Rechenungenauigkeit. In realen Anwendungen ist eine vollständige Übereinstimmung jedoch nicht möglich. Die Gründe dafür reichen von unterschiedlichen Programmiersprachen über leicht unterschiedliche Implementierungen bis zu unterschiedlichen Unterfunktionen und vieles mehr. Das Ziel ist es zu zeigen, wo und wie groß die Unterschiede zwischen diesen beiden Modulen sind. Wir zeigen Bespiele für unterschiedliche Aspekte, die aus unterschiedliche Kategorien der Unsicherheiten stammen. Dies inkludiert Beispiele zu Unsicherheiten aus den numerischen Methoden, Implementierungen oder Input. Wir zeigen, dass wir die Unterschiede für gering genug halten, um alle zukünftigen Entwicklungen mit com1DFA als Basisreferenz durchzuführen.

Für den beigefügten Bericht sind folgende Versionen verwendet worden:

- com1DFA: github commit 32408c5fc74dd8b (26. Jan. 2022)
- com1DFAOrig: git.avaframe.org commit c771be02e3c771b

Es ist zu beachten, dass com1DFAOrig, der extrahierte DFA-Kernel von SamosAT, verwendet wird und nicht SamosAT selbst. Es gibt bereits Unterschiede in den Ergebnissen zwischen SamosAT und com1DFAOrig (siehe auch github issue number 109). Es wird vermutet, dass dies ein Compiler-Problem ist, die genaue Ursache ist noch nicht geklärt. Auch wenn diese Unterschiede erkennbar sind, so sind sie doch gering im Vergleich zu den nächsten Problemen.

Es gibt zwei Vorbehalte bezüglich des direkten Vergleichs zwischen com1DFAOrig und com1DFA:

1. Unsere Untersuchungen haben gezeigt, dass die anfängliche Partikelverteilung eine wichtige Rolle für die Simulationsergebnisse spielt. Selbst eine einfache Änderung wie die des Seeds der Zufallszahlen, das in com1DFAOrig verwendet wird, führt zu Unterschieden im Auslauf. Um Unterschiede zu vermeiden, die durch unterschiedliche Zufallszahlengeneratoren in den verschiedenen Programmiersprachen c++ und python entstehen, haben wir uns entschieden, folgendes Verfahren zum Vergleich zu verwenden:
   - Modifikation von com1DFAOrig, um die anfängliche Partikelverteilung in eine Datei auszugeben
   - com1DFAOrig ausführen, um Ergebnisse und die anfängliche Partikelverteilung zu erhalten
   - Einlesen und Verwendung dieser Verteilung in com1DFA, wodurch die Partikelverteilung der com1DFA python-Implementierung komplett umgangen wird. Allerdings ermöglicht dies einen besseren Vergleich der numerischen Implementierung selbst.

   Nebenbei bemerkt: Das Problem der Abhängigkeit vom Seed der Zufallszahlen besteht auch in com1DFAOrig (und SamosAT). Wir haben bereits alternative Ansätze untersucht, aber bisher nur inkrementelle Verbesserungen gefunden, die das Problem nicht wirklich lösen.

2. Ein zweites Problem, das dem Vorhergehenden ähnelt, ist die Behandlung der digitalen Höhenmodelle (DEM). Genauer gesagt geht es um die Handhabung der Zellgrößeninformationen aus dem ASCII-Header. Wenn die Zellgröße auf etwas anderes als die in den beiden Modulen für die Numerik verwendeten 5 m gesetzt wird, findet ein Remeshing des eingegebenen DEMs statt. Aufgrund der leicht unterschiedlichen Implementierungen führt dies zu geringfügig unterschiedlichen Simulations-DEMs. Diese kleine Änderung beeinflusst dann die Platzierung der Partikel und führt zu demselben Problem wie im vorherigen Punkt. Wir haben einige Testfälle,

bei denen wir gezielt eine andere Zellgröße als 5 Meter (4.9XX) verwenden, mit der ausdrücklichen Absicht, das Remeshing-Verhalten testen zu können. Um einen besseren Vergleich zu ermöglichen, haben wir für diesen Bericht alle DEMs auf eine Auflösung von 5 m zurückgesetzt, um ein Remeshing zu vermeiden.

Im Folgenden zeigen wir die Hauptunterschiede anhand von bestimmten Fallstudien mit jeweils nur einer Abbildung. Weitere Abbildungen zu jeder Fallstudie sind in den angefügten Berichten zu finden.

Wellenmuster / Randbereiche

Im Bereich von Wellenmustern (die ihren Ursprung nicht in der Topographie haben) kann es zu sichtbaren Unterschieden in den Lawinensimulationen kommen. Das kann man zum Beispiel in der maximalen Fließtiefe der kanalisierten Helix-Lawine in Abbildung 1 sehen. An Stellen mit Wellen kommt es auch zu den grössten Unterschieden zwischen com1DFAOrig und com1DFA, wobei beide Berechnungsmodule ähnliche Muster produzieren. Aber wenn die Wellen um ein oder zwei Gitterzellen verschoben sind, kommt es dort aufgrund der starken Gradienten zu großen Unterschieden. Das selbe Problem tritt an den Rändern der Lawine auf, auch dort verursacht ein Versatz von wenigen Gitterzellen große Unterschiede. Dieser Effekt ist zum Beispiel auch in den Simulationen von avaHit und avaHockeySmall zu beobachten.

Steile Bereiche

Im mittleren Bereich der Lawine von avaGar mit Anbruchgebiet relGar2 sind deutliche Unterschiede in den Simulationsergebnissen zu erkennen (siehe Abbildung 2). In diesem Bereich der Lawine gibt es Felsabbrüche, dieses sehr steile Gelände bedeutet eine starke Krümmung. Bei starker Krümmung verstärken sich auch die Unsicherheiten die mit den Krümmungstermen im Modell verbunden sind. Zusätzlich dazu müssen die Partikel in jedem Zeitschritt zurück auf die Geländeoberfläche projiziert werden. Das ist wichtig, da es durch die Bewegung pro Zeitschritt dazu kommen kann, dass sich ein Partikel nicht mehr an der Geländeoberfläche befindet aber etwas darunter (selten) oder darüber. Im Modell ist es nicht möglich, dass sich Partikel von der Geländeoberfläche “lösen“. Es gibt unterschiedliche Möglichkeiten die Partikel zurück auf die Geländeoberfläche zu projizieren, wie z.B. in Richtung der Schwerkraft, orthognal zur Geländeoberfläche, etc. und auch hier gilt, je steiler das Gelände, umso größer werden die Unterschiede zwischen den einzelnen Methoden.

Die Unterschiede scheinen noch ausgeprägter zu sein wenn Coulomb Reibung mit resultierenden höheren Geschwindigkeiten verwendet wird. Das heisst, je größer der Zeitschritt, desto wichtiger ist der Projektionsterm.

Allerdings sind die Ergebnisse im Auslaufbereich trotz der Abweichung im mittleren Teil nahezu identisch!

Kleine Seitenarme

Abbildung 3 zeigt die maximale Fließtiefe der Gar-Lawine für das relGar Anbruchgebiet. Für die Simulationsergebnisse von com1DFAOrig ist ein kleiner Seitenarm im oberen Bereich der Lawine zu erkennen. Die Entwicklung solcher Seitenarme haben wir nur in sehr wenigen Simulationen beobachtet, aber immer mit derselben Charakteristik: einige Partikel fließen auf der ’falschen’ Seite einer Geländekante talwärts. Dieses Problem tritt auch manchmal in Com1DFA Simulationen auf. Zudem kann man diese Seitenarme normalerweise mit einer kleinen Anpassung des Anbruchgebiets verhindern. Daher führen wir deren Entwicklung auf kleine Unterschiede in den Anfangspositionen der Partikel zurück und sehen darin kein größeres zugrunde liegendes Problem. Da es dadurch im Auslaufgebiet zu praktisch keinen Unterschieden kommt, halten wir es für akzeptabel.

3.1 Schlussfolgerung und Empfehlung

Betrachtet man die beigefügten Berichte für die Null-, Entrainment- und Widerstandssimulationen in Kombination mit den Abschnitten in diesem Bericht, so zeigt sich, dass die Unterschiede der

Simulationen oft nicht mit dem Auge erkennbar sind, wenn man die vollständigen Simulationen betrachtet. Sowohl Widerstand als auch Entrainmentsimulation zeigen Unterschiede in der Größenordnung von 0.5 Prozent für die Gesamtmasse (2,5 Prozent für die Entrainment-Masse) mit leichten Unterschieden in in der Mitte der Lawinenbahn, jedoch ohne Auswirkungen auf den Auslaufbereich. Der Nullsimulationsbericht, der alle realen Lawinen-Testfälle enthält, zeigt ebenfalls visuell identische Ergebnisse für alle Spitzenvariablen.

Somit: trotz der zuvor beschriebenen Unterschiede und Vorbehalte, unter Betrachtung aller Beispielsimulationen zeigt sich, dass die generellen Muster sehr ähnlich, wenn nicht identisch sind. Da die absoluten Unterschiede sich in einer akzeptablen Größenordnung abspielen und deren Ursprung nachvollziehbar ist, schließen wir daraus, dass com1DFAOrig und com1DFA ausreichend ähnliche Ergebnisse liefern. Die hier beschriebenen Unterschiede liegen innerhalb der Modellunsicherheit, siehe dazu ein detailliertes Beispiel im Abschnitt 7.3.

Wir empfehlen, dass die Berechnungsmodule weiterhin Seite an Seite von praktischen Anwendern im Rahmen derer operationallen Arbeit getestet werden. Dadurch soll sichergestellt werden, dass weitere Probleme und Schwierigkeiten die in der praktischen Anwendung eine Rolle spielen beobachtet werden können.

4 Methods and test cases

See the testing section of the documentation https://docs.avaframe.org/en/latest/testing.html for a description of all test cases used in the attached report. Here are some notes on the avalanches used later in this document:

- avaInclinedPlane and avaInclinedPlaneDiag: useful for having constant slope
- avaParabola: curvature only in one direction, main flow direction aligned with raster
- avaHelix and avaHelixChannel with identical release: curvature in two directions, flow direction not aligned with raster
- avaWog, avaKot: with a set cellSize to 5m and xllcenter, yllcenter to mulitple of cellSize

Figures in this document are concentrated on displaying differences between two simulations, one used as reference (ref), one called simulation (sim). The variables displayed in most cases are one of the peak flow parameter variables, i.e. peak pressure, peak flow depth and peak velocity. Differences are calculated raster based since output results are aligned on the same grid. Statistics for the differences are also based on the raster results, however sometimes AIMEC transformations are used (see documentation for more info), with manual avalanche paths. Insets on figures show either boxplots or cumulative distribution plots. Analysis is only done where one or both results have values. Probability maps combine multiple simulations, all weighted the same, showing fractions: ie. fraction = 1.0 means all simulations hit this pixel, 0.0 means no simulation covers it. We did not use an absolute acceptable error range, this is based on expert opinion looking at minimum/maximum values and standard deviations. Errors close to the float precision are seen as less significant and values are judged in relation to their absolute values.

5 Performance (computing time, RAM usage)

In this section we discuss the computational performance, i.e. information about compute time and RAM usage. A few remarks regarding the interpretation of these numbers:

- all code was run sequential, no parallelization is used.

- the default compiler (gcc) optimization level is used (-O).
- so far no efforts to improve com1DFA performance took place
- the time saving in case of using the QGis connector to start com1DFA (i.e. not having to transfer data between QGis and SamosAT; in operational applications) is not considered.
- experiments were run on 26th January 2022 with the current master.

Table 1 shows the computation times for all experiments in the attached reports. Times are rounded to the full second. The median of the speed factor is 1.62, the average factor is 1.7. This means com1DFA takes about 60-70 percent more computing time than com1DFAOrig.

Regarding RAM usage output of GNU Time (/usr/bin/time –verbose) is a follows, with the 'Maximum resident . . . ' being RAM usage:

```
Command being timed: "python com1DFAOrig/runCom1DFAOrig.py"
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:25.71
Maximum resident set size (kbytes): 191176
File system outputs: 357712

Command being timed: "python runCom1DFA.py"
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:48.54
Maximum resident set size (kbytes): 275940
File system outputs: 20968
```

In this case com1DFAOrig uses about 190 Mb of RAM, whereas com1DFA uses about 275 Mb. The difference is that com1DFA keeps all data in RAM, whereas com1DFAOrig write to files right away freeing memory. This can be seen in the *File system outputs* information: com1DFAOrig writes about 350-thousand times versus 20-thousand time for com1DFA. As of now this is of no concern, but once more simulations are run at the same time, this will be needed to be optimized.

Disk usage of the final results is about the same for the peak files. All other output (time steps, images etc) depends hugely on the chosen output options. This holds true for both com1DFA and com1DFAOrig, so no comparison was done.

Simulation / Computation time ->	com1DFAOrig [s]	com1DFA [s]	Factor
release1FP_null_dfa_c26d996278910	6	7	1.22
relGar2_null_dfa_74c2ab437d	115	165	1.43
relGar_null_dfa_97985220a6	23	33	1.45
relGar6_null_dfa_809e2c34d6	26	39	1.48
relMal1to3_null_dfa_046aa835ad	358	532	1.49
relHit_null_dfa_492d74a9a9	100	149	1.49
relWog_null_dfa_0ce7e1912b	93	141	1.51
relWog_ent_dfa_9ddad053df	106	167	1.57
release1HX_null_dfa_ec32971d1a670	60	95	1.59
relHit_ent_dfa_05638ab740	149	239	1.60
release1HX_ent_dfa_e5743f0dd3	59	95	1.62
relMal1to3_ent_dfa_0817f3a118	488	795	1.63
release1HX_null_dfa_9201630216750	35	59	1.70
release1PF_null_dfa_61be8879b7981	35	61	1.72
relKot_null_dfa_f78102228e	15	27	1.76
release1HS_null_dfa_7e525483c51052	54	99	1.83
release1PF_res_dfa_3adc0d5d1c	35	65	1.86
relKot_res_dfa_9a0c4083ac	15	28	1.88
relKot_ent_dfa_701aade463	15	29	1.92
relAlr_ent_dfa_b3866fb76d	36	73	2.05
relAlr_null_dfa_0ae508d0cf	32	70	2.21
release1BL_null_dfa_c5e8990a42830	61	150	2.44

Table 1: Computation times in [s] for the experiments in the attached reports. The median of the speed factor is 1.62, average 1.7. I.e. com1DFA is about 60-70 percent slower.

6 Flow model components

In these tests we took a more targeted approach as compared to using the full standard test cases in the reports. We started from only simulating a simple avalanche for the simplest model configuration and then subsequently added more processes / components in order to identify when differences first develop. The aim of this is to get an 'impression/feeling' for the model behavior and not an exhaustive investigation of each component. It allowed us to use a more targeted approach for the development issues that needed to be tackled.

6.1 Coulomb friction / Curvature

- avalanches: parabola and inclined plane (slope direction is not aligned with the DEM raster), release area: standard release and initial particle locations imported from SamosAT
- Coulomb friction only (no lateral force, no artificial viscosity)

Looking at differences in the parabola case, figure 4 right panel, noticeable effects show up in the bend part of the topography, i.e. where curvature is present. Simply removing curvature term in the acceleration does not seem to improve things, see figure 5 left panel. However decreasing the time step from 0.1s to 0.01s improves the results, showing smaller difference values, see 5 right panel. Adding curvature back again but this time with the small time step leads to the results shown in figure 6.

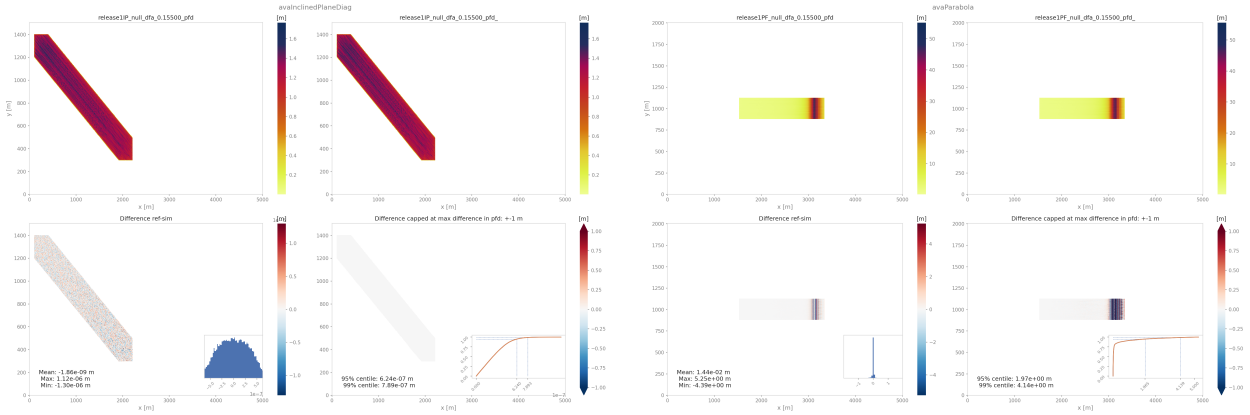

Figure 4: Simulation with Coulomb friction, Curvature, no pressure lateral nor artificial viscosity (dt=0.1s) down two idealized topographies: an inclined plane on the left (slope running from top left to bottom right of the figure) and a parabolic slope on the right.

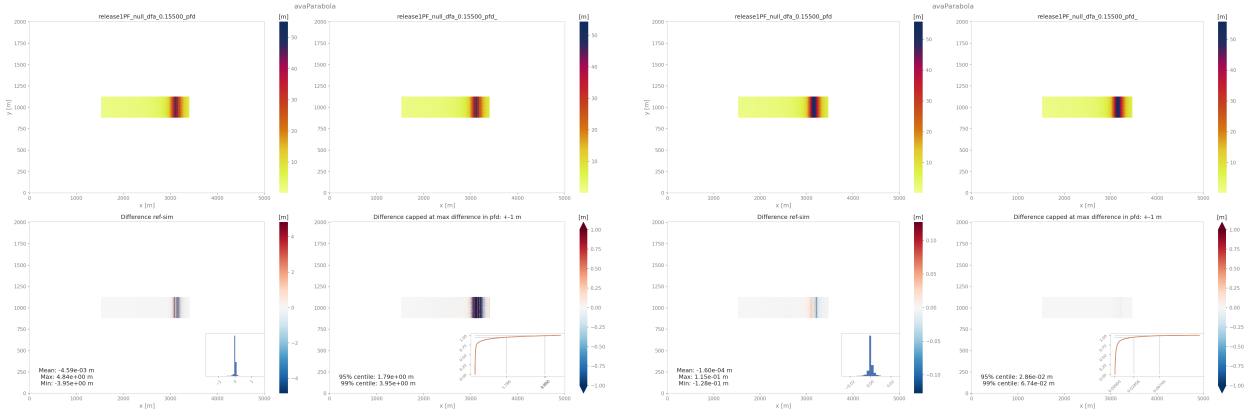

Figure 5: Simulation on a parabolic slope (idealized topography) with no curvature acceleration term. In the left panel, dt=0.1s and decreasing the time step to dt=0.01s in the right panel.

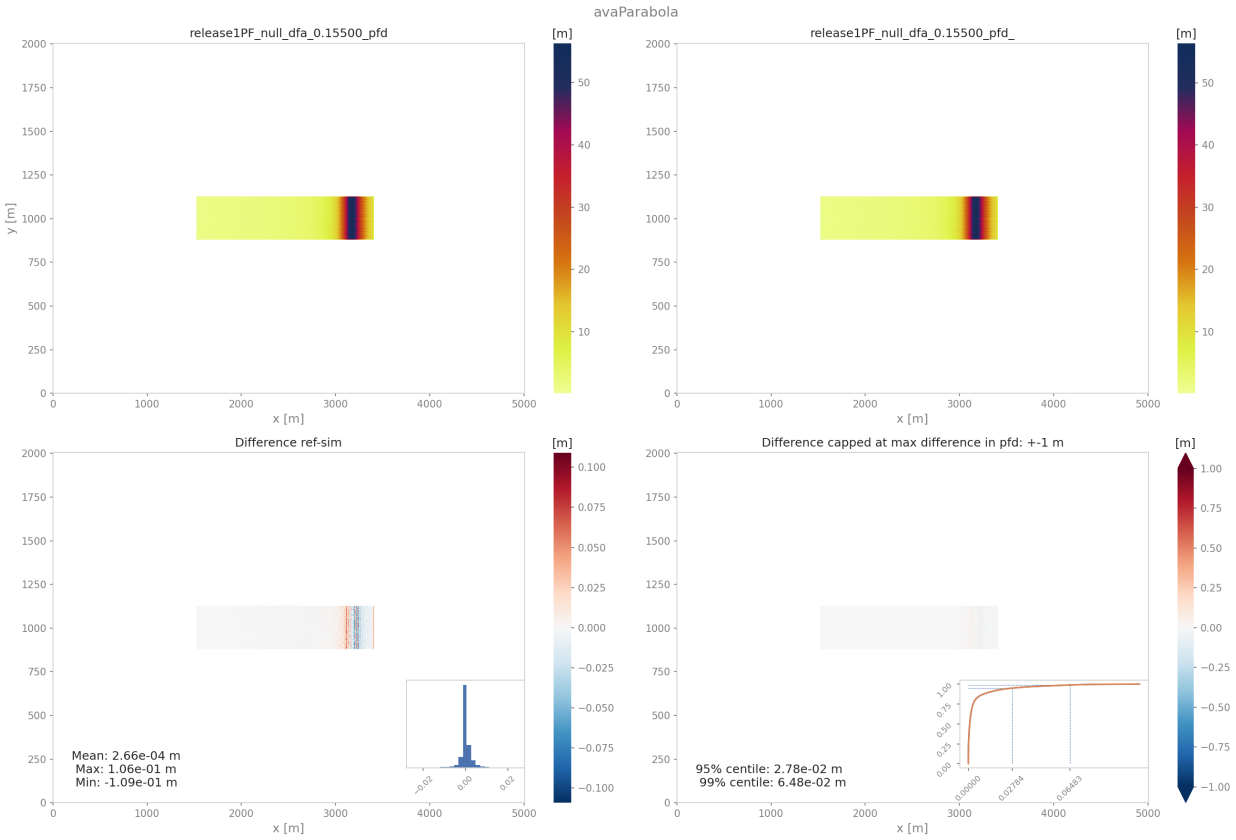

Figure 6: Simulation on a parabolic slope (idealized topography) with Coulomb friction, Curvature, no lateral forces nor artificial viscosity for a time step dt=0.01s

6.2 SamosAT friction, no pressure gradients

- avalanches: inclined plane (this time, slope direction is aligned with the dem raster), release area: standard release imported from SamosAT
- SamosAT friction only (no lateral force, no artificial viscosity)

The standard setup for this is shown in figure 7 left panel. Experiments with shorter time steps (0.025s; figure 7 right panel) show no significant improvement.

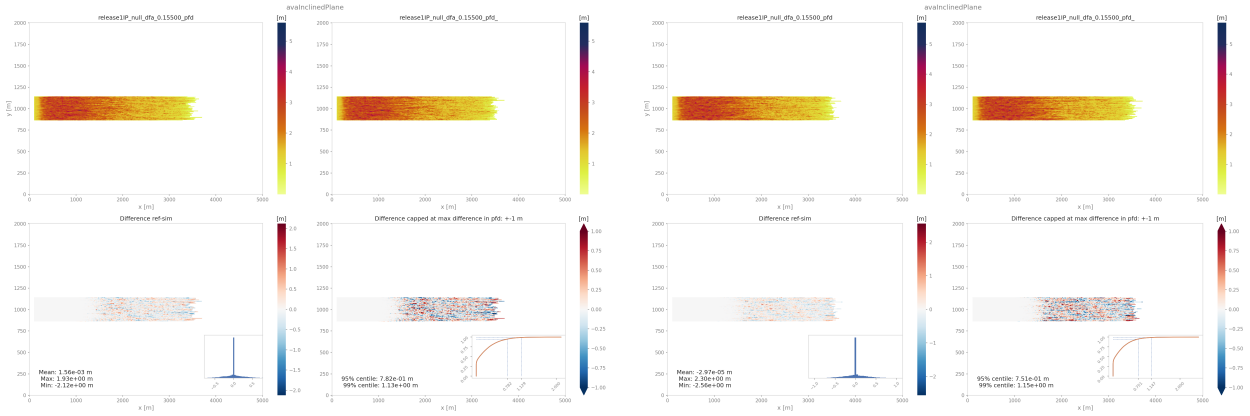

Figure 7: Simulation on an inclined plane (idealized topography) with SamosAT friction, Curvature, no lateral forces nor artificial viscosity for a time step dt=0.1s in the left panel and dt=0.025s in the right one.

6.3 Coulomb friction, pressure gradients, artificial viscosity

- avalanche: inclined plane, release area: standard release areas
- Coulomb friction with lateral force and artificial viscosity

The standard setup for this is shown in figure 8 left panel. Decreasing the time step to 0.01s improves the results (factor 10 on 95 percent values), see figure 8 right panel.

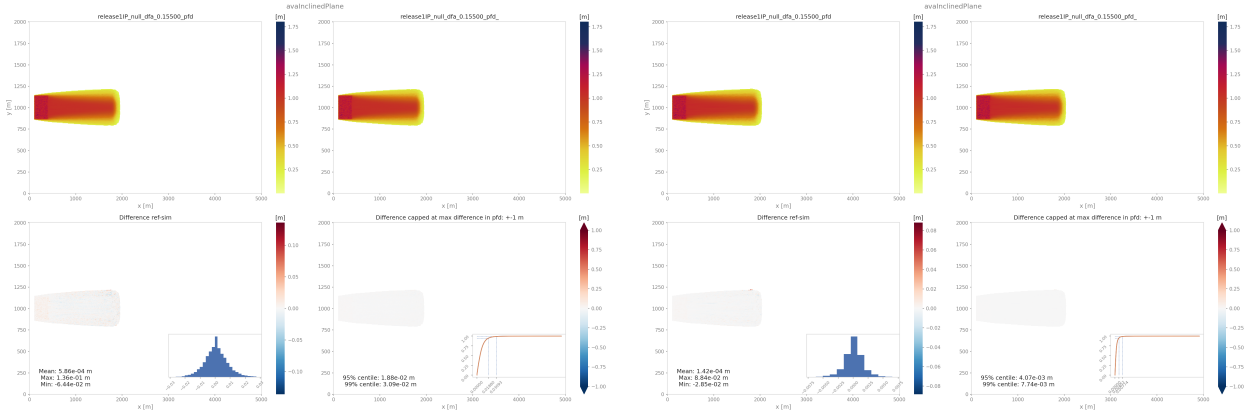

Figure 8: Simulation on an inclined plane (idealized topography) with Coulomb friction, Curvature, lateral forces and artificial viscosity for a time step dt=0.1s in the left panel and dt=0.01s in the right one.

6.4 SamosAT friction, pressure gradients, artificial viscosity

- avalanche: inclined plane, release area: standard

- SamosAT friction, lateral force and artificial viscosity

The standard setup for this is shown in figure 9 left panel. Decreasing the time step from 0.1s to 0.025s improves the results (for example the 95 percent values). It also pushes the point where the first significant differences appear from 500m to 1000m down slope (see figure 9 right panel).

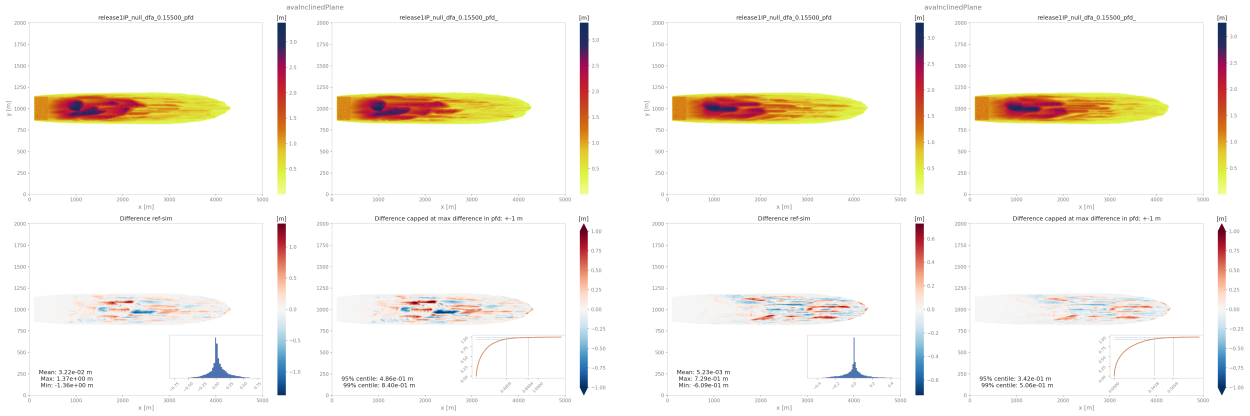

Figure 9: Simulation on an inclined plane (idealized topography) with SamosAT friction, Curvature, lateral forces and artificial viscosity for a time step dt=0.1s in the left panel and dt=0.025s in the right one.

6.5 Entrainment

As examples for differences related to entrainment, we show the test cases *avaHockeyChannel*, *avaHelixChannel* and *avaAlr*. Figures 10 and 11 show entrained mass versus time plots for the idealized avalanches. Figure 12 show the same for the real avalanche avaAlr. Figure 13 also shows the difference plot for this avalanche.

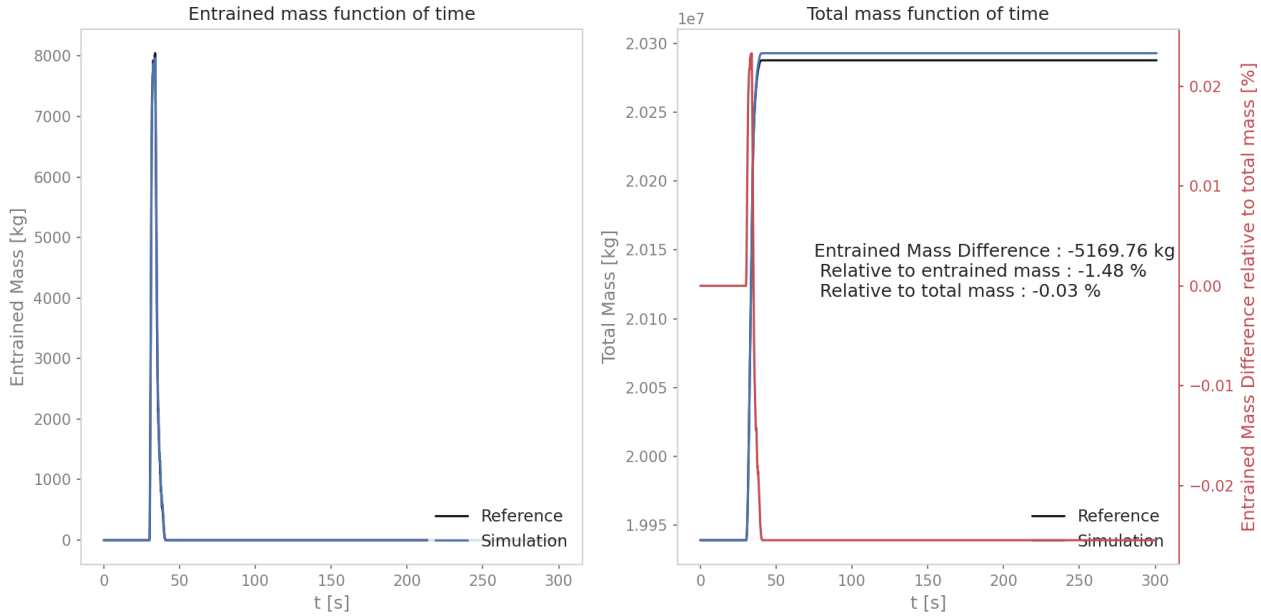

Figure 10: Entrainment mass over time for avaHockeyChannel (idealized topography). In the left panel, lines are almost identical.

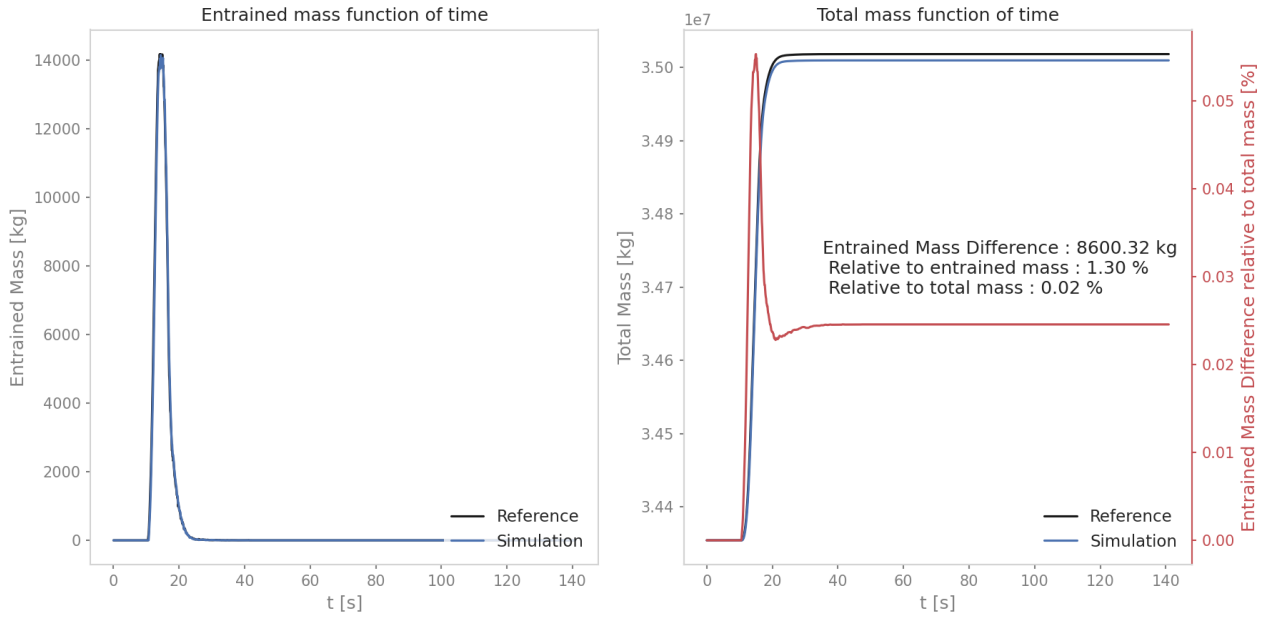

Figure 11: Entrainment mass over time for avaHelixChannel (idealized topography). In the left panel, lines are almost identical.

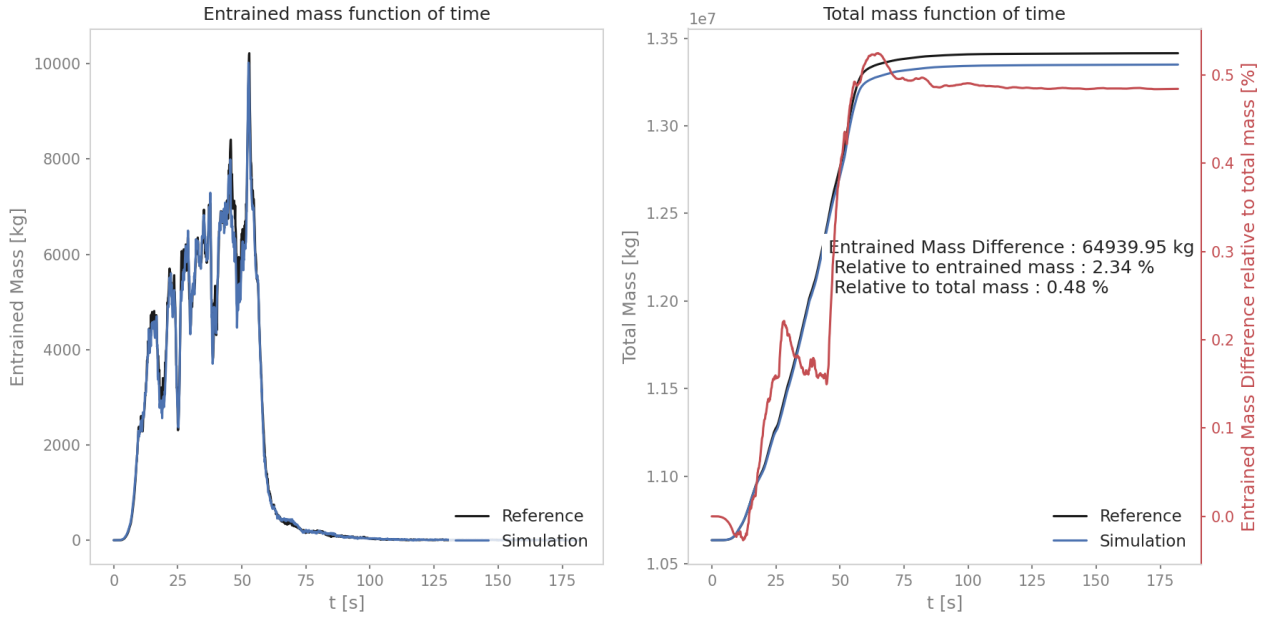

Figure 12: Entrainment mass over time for avaAlr (real topography)

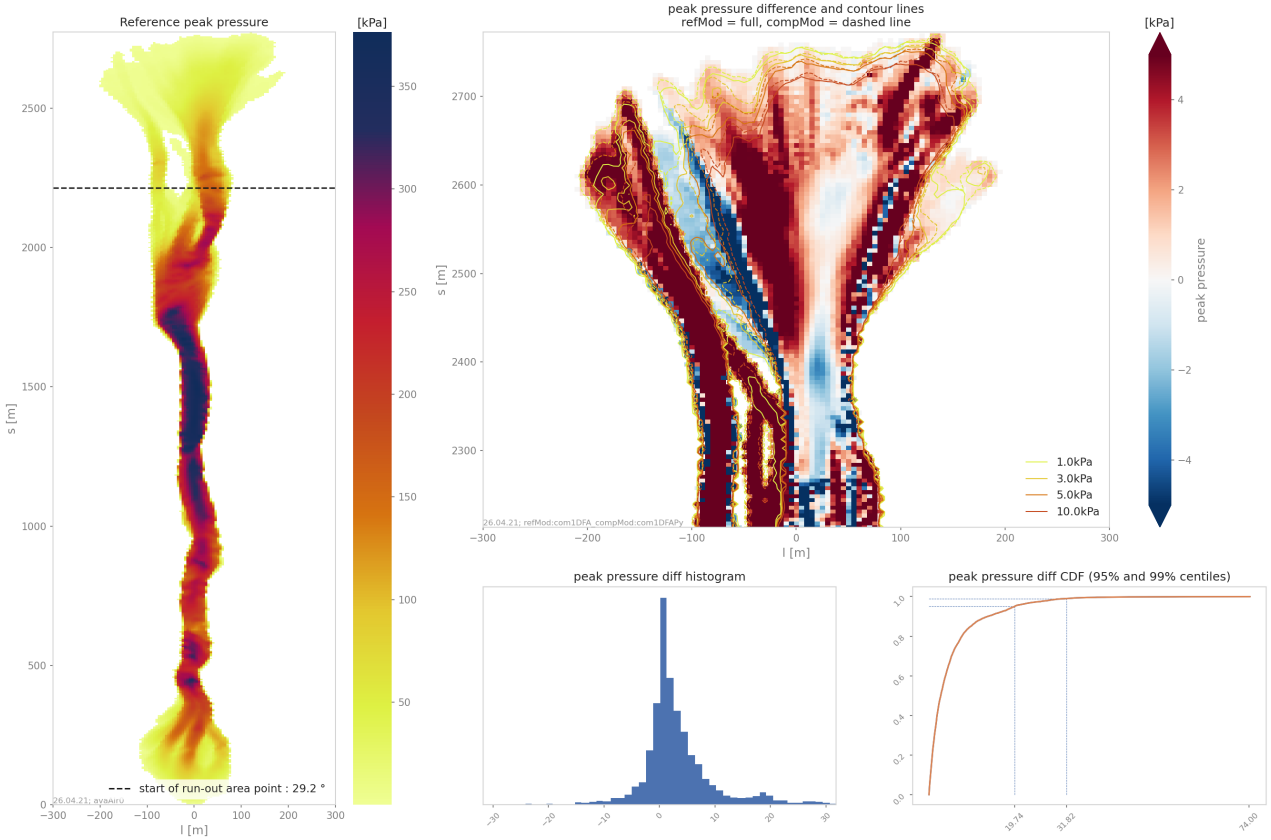

Figure 13: Difference plots (created with AIMEC) for com1DFAOrig vs com1DFA with entrainment enabled for avaAlr (real topography)

6.6 Resistance

As examples for differences related to resistance, we show the test cases *avaParabola* and *avaKot*. Figures 14 and 15 shows distributions for peak variables along the path for com1DFAOrig (reference) and com1DFA (simulations). Figure 16 shows a peak pressure comparison for avaKot.

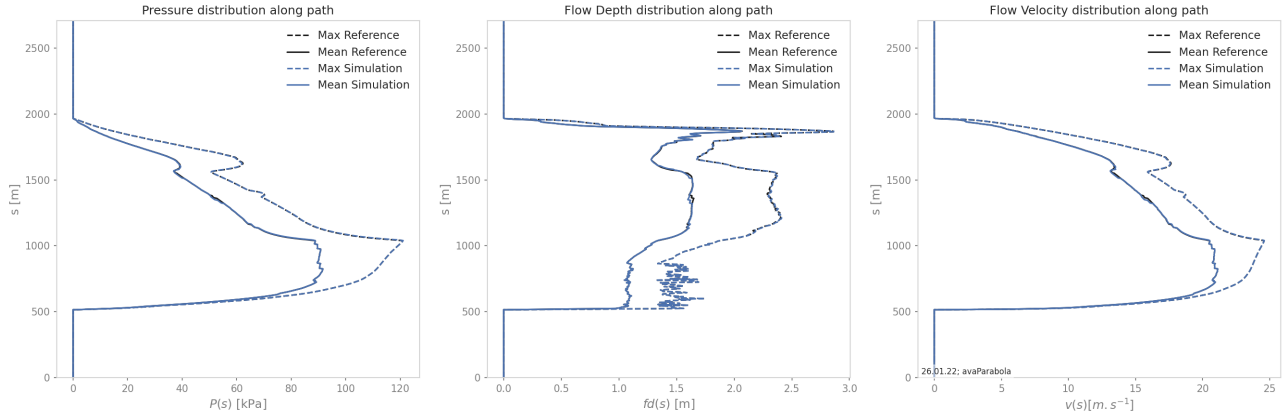

Figure 14: Difference plots for com1DFAOrig (reference) vs com1DFA (simulation) with resistance enabled for avaParabola (idealized topography). Blue and black lines are almost identical.

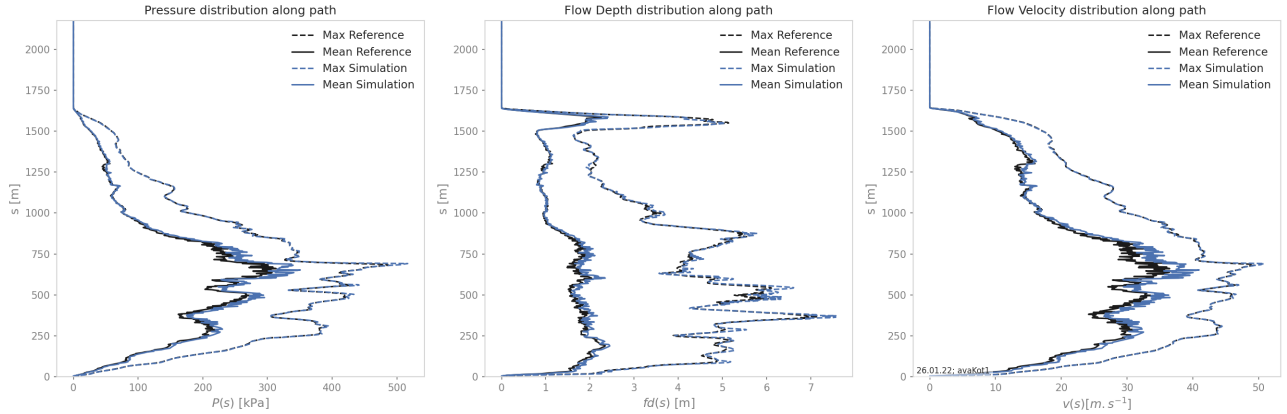

Figure 15: Difference plots for com1DFAOrig (reference) vs com1DFA (simulation) with resistance enabled for avaKot (real topography).

7 Numerical components

7.1 Changing time step

Changing the time step affects significantly the results. The comparison in this section is done between com1DFAOrig with dt=0.1s and com1DFAOrig with dt=0.025s to show the range of differences simply arising from different time steps Figure 17 left panel shows the comparison of initial condition for dt=0.1s and dt=0.025s. The right panel shows the comparison after 400s.

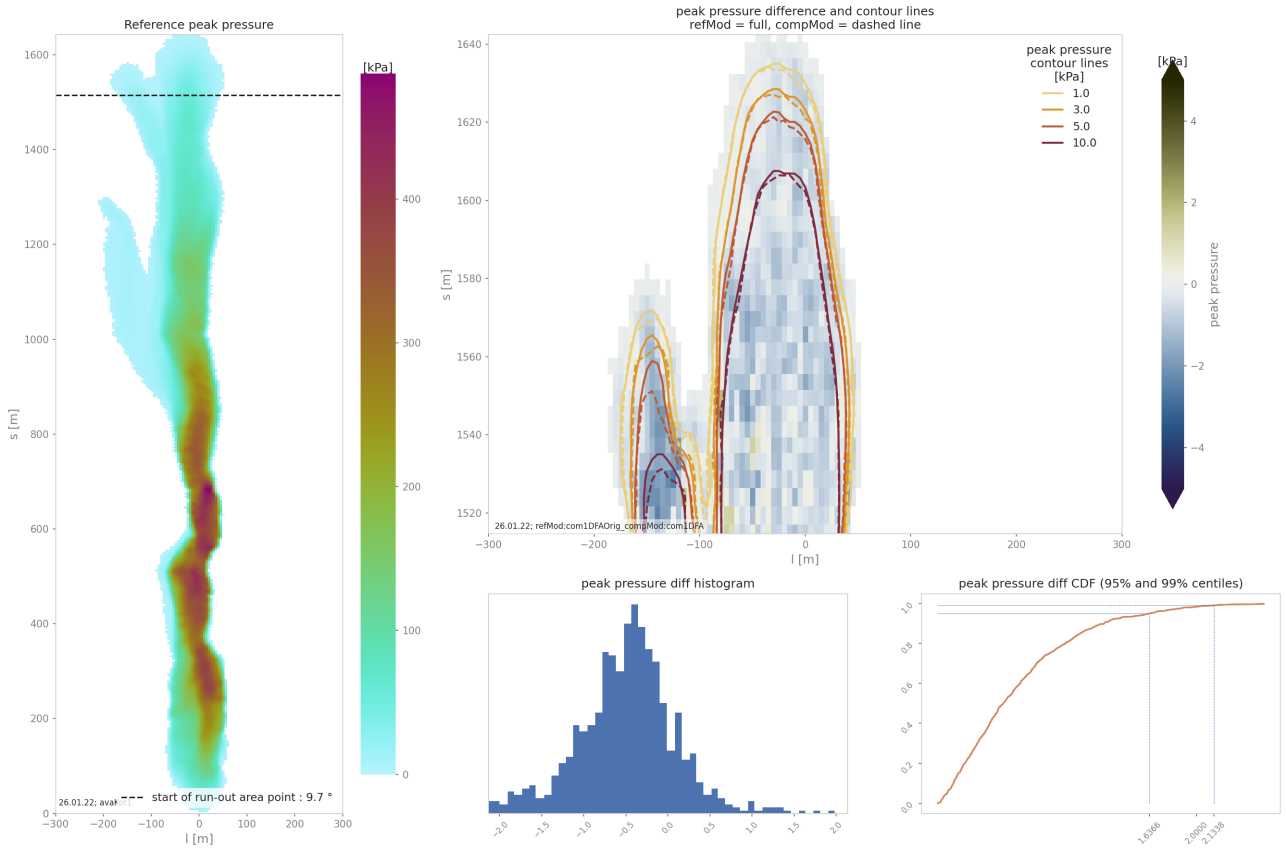

Figure 16: Difference plots for com1DFAOrig (reference) vs com1DFA (simulation) with resistance enabled for avaKot (real topography).

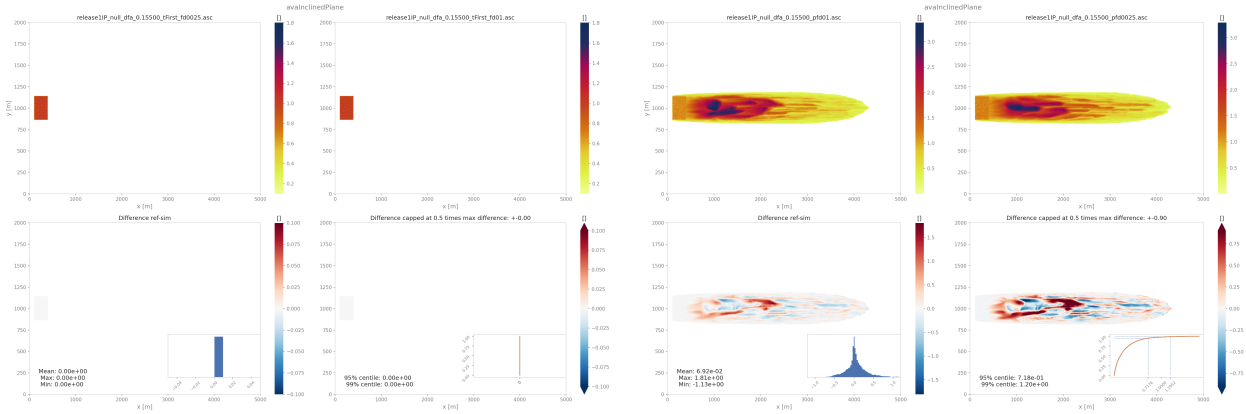

Figure 17: Difference of Com1DFAOrig peak flow depth [m] for avaInclinedPlane (idealized topography) with time step of 0.025s and 0.1s at the start of the simulations (initial condition - left panel) and after 400s of simulation (right panel). Lower left panels show the full range of differences where the right lower panel is capped to 50% of the max difference to give more detail on the difference pattern.

7.2 Single vs double precision

In the com1DFAOrig code variables are defined as floats (single precision; $1e-6$) or double (double precision; $1e-12$). The c++ code is compiled with gcc compiler. We compared two versions of the c++ code, where the floats are turned into doubles and then recompiled the code. Already this had an impact on the simulation results - however the magnitude is lower compared to the difference com1DFAOrig vs com1DFA. Figure 18 shows the differences for the inclined plane between single and double precision version (time step dt=0.1s).

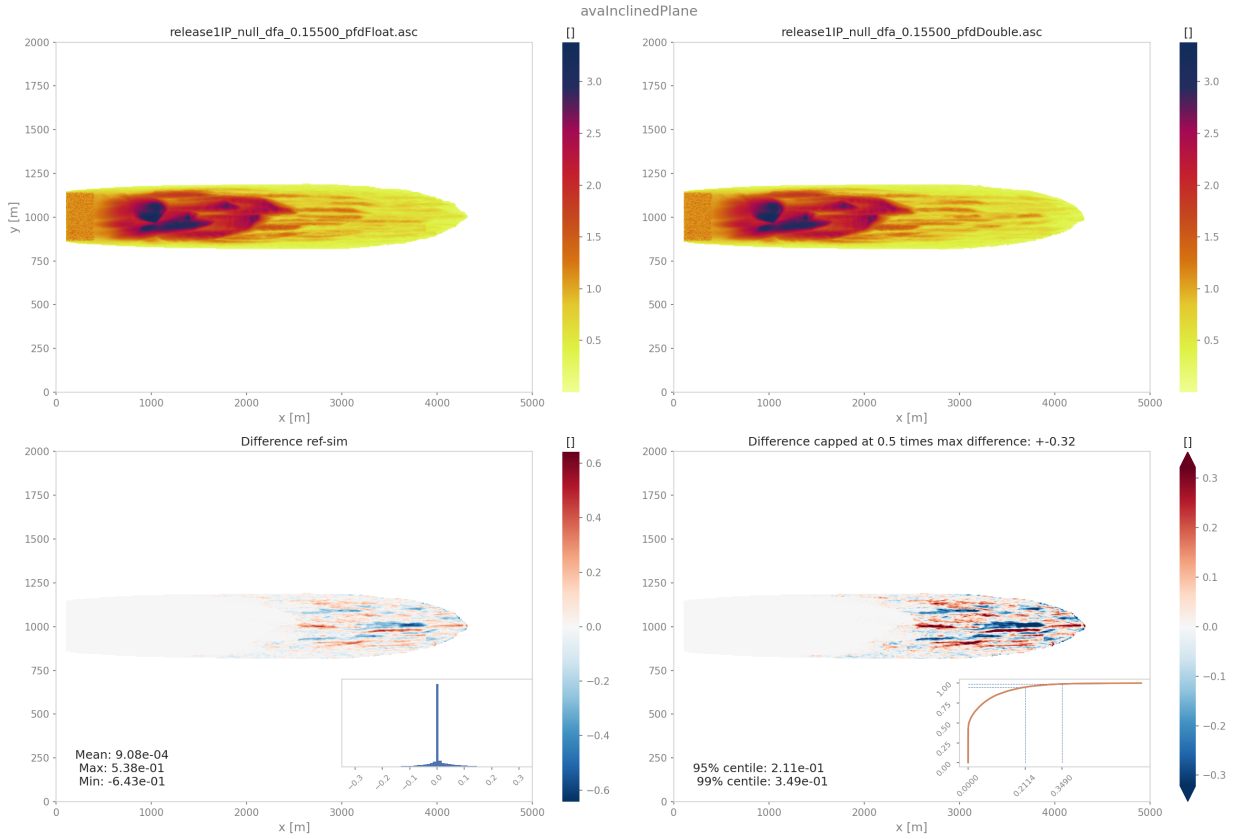

Figure 18: Difference of Com1DFAOrig peak flow depth [m] using the standard implementation (ref) with doubles insteand of floats (sim) for the avaInclinedPlane (idealized topography).

7.3 Initial Particle location

In order to initialise the simulations, the release area is represented by a number of particles. These are located randomly within each cell. For this random location generation, a so called random seed is set. This way the initial particle distribution for a given release area, DEM and number of particles is reproducible.

However, if a different seed is used, the random locations of the particles within a cell will differ. This can have a substantial effect on the simulation results. To demonstrate the differences introduced by choosing a different seed value, we perform simulations with the following values for the seed: 12345, 29374, 87132, 42958, 38264, 65924, 75683, 54975, 92428, 72639, 27, 563, 5, 1728, 1. For com1DFAOrig, this is used to initialize the random number generator in SW_Workspace.cpp line 2779. In case of com1DFA, the seed is a parameter that can be set in the module's configuration file (com1DFACfg.ini).

For these tests we use avaAlr (with cellSize set to 5m, and lowerleftcenter coordinate set to a multiple of 5m) and avaParabola. A 'null' simulation setup for is used for both tests.

In order to demonstrate the effect of choosing a different seed value, we first show probability maps of peak pressure (threshold 1kPa) for both, com1DFAOrig and com1DFA, individually (see figs. 19, 20, 21, 22).

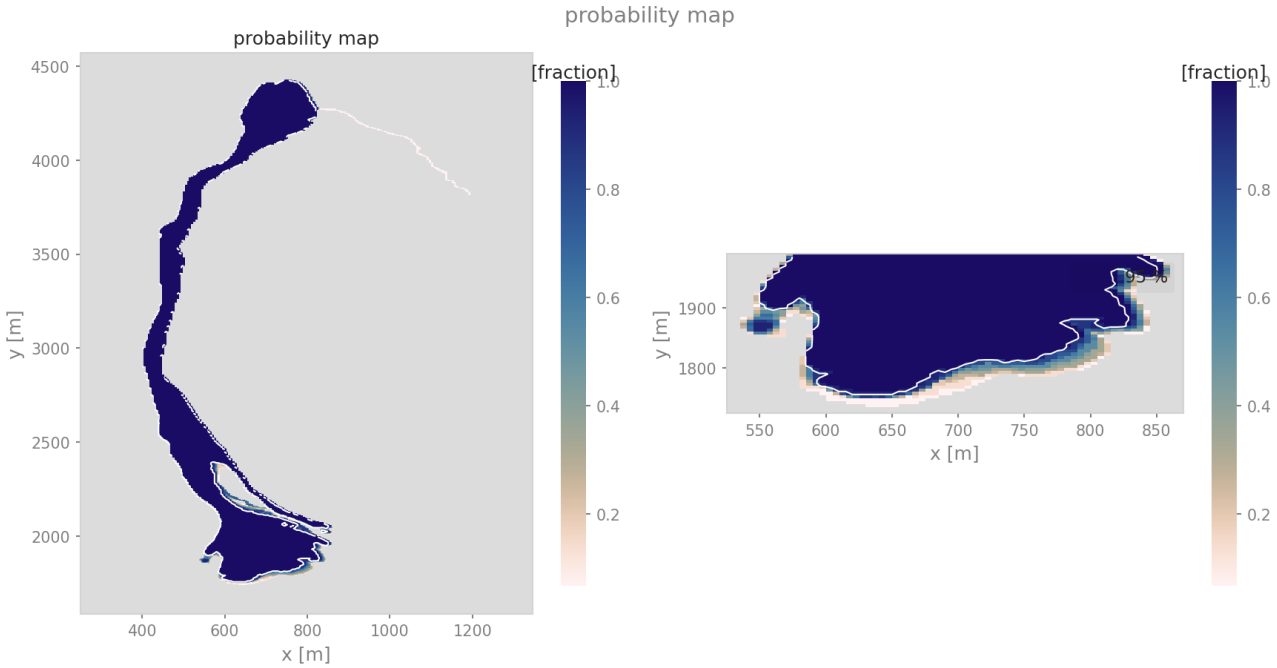

Figure 19: Probability map of com1DFAOrig peak pressure for avaAlr (real topography) for 15 simulations with different seed values. The fraction (from 0-1, referring to 0-100 %) of simulations with peak pressure exceeding 1kPa is shown, including the 95% contour line. Right panel shows a zoom into the lower part of the avalanche.

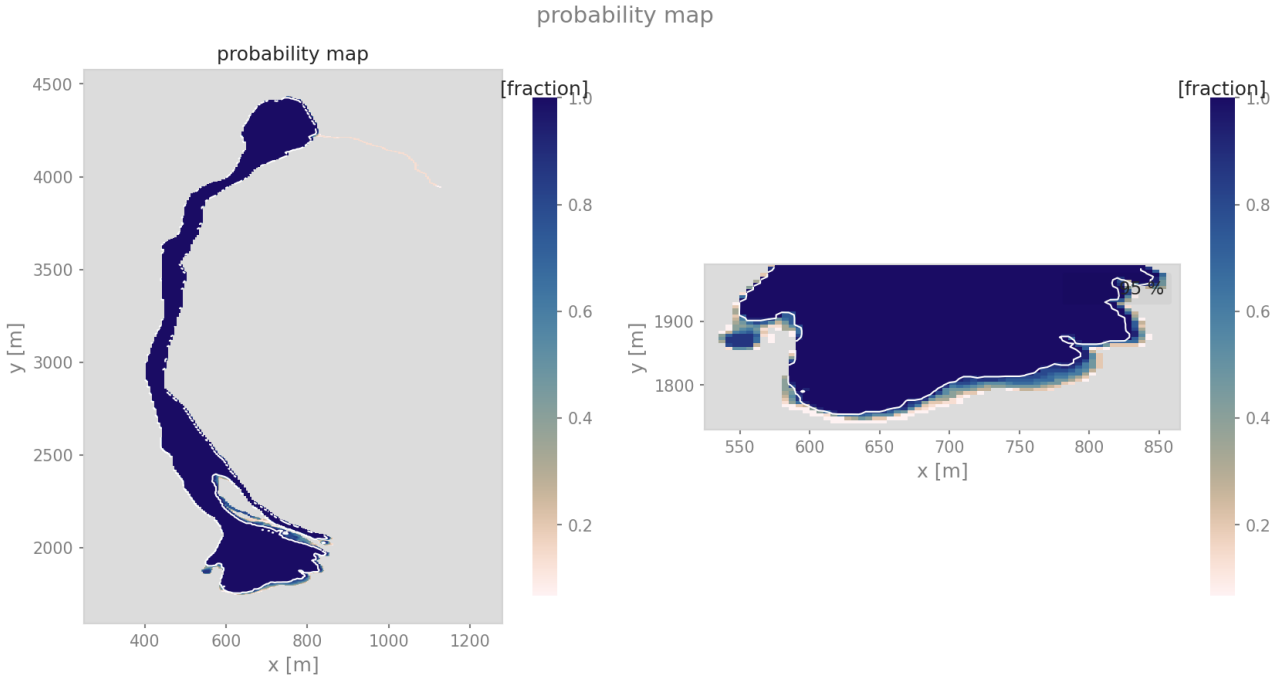

Figure 20: Probability map of com1DFA peak pressure (threshold 1kPa) for avaAlr (real topography) for 15 simulations with different seed values. The fraction (from 0-1, referring to 0-100 %) of simulations with peak pressure exceeding 1kPa is shown, including the 95% contour line. Right panel shows a zoom into the lower part of the avalanche.

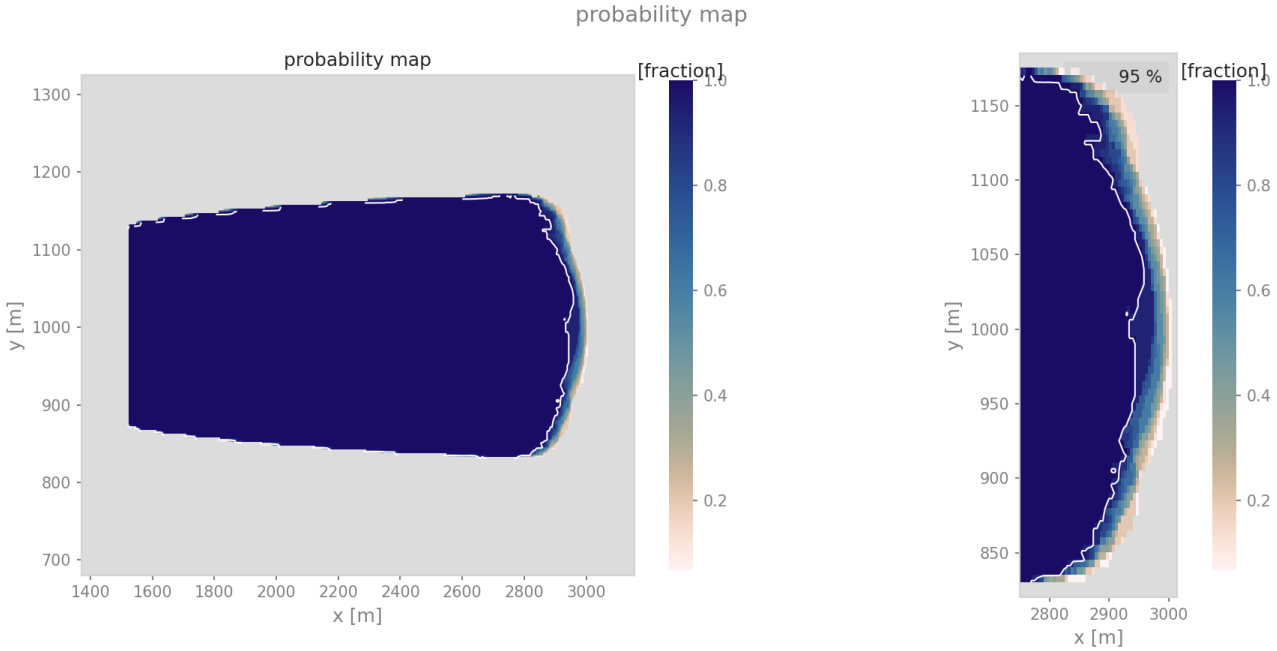

Figure 21: Probability map of com1DFAOrig peak pressure (threshold 1kPa) for avaParabola (idealized topography) for 15 simulations with different seed values. The fraction (from 0-1, referring to 0-100 %) of simulations with peak pressure exceeding 1kPa is shown, including the 95% contour line. Right panel shows a zoom into the lower part of the avalanche.

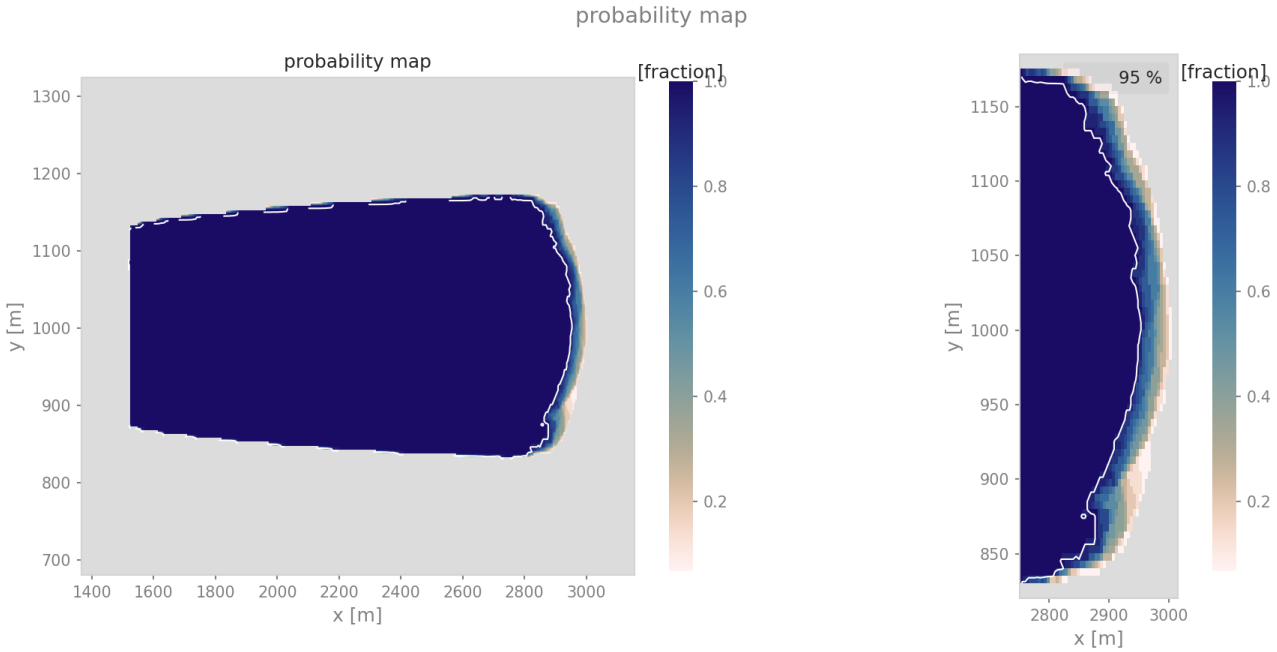

Figure 22: Probability map of com1DFA peak pressure (threshold 1kPa) for avaParabola (idealized topographhly) for 15 simulations with different seed values. The fraction (from 0-1, referring to 0-100 %) of simulations with peak pressure exceeding 1kPa is shown, including the 95% contour line. Right panel shows a zoom into the lower part of the avalanche.

We also analyse the set of simulations along the avalanche path in order to be able to reduce the avalanche runout to a scalar value and compare in between simulations. In tables 2, 3, 4, 5, 6, statistical measures of the scalar runout as well as maximum cross profile values of peak pressure, peak flow depth and peak flow velocity are shown. Tables are based on the 15 simulations with different seed values for each computational module.

	sRunout[m]	std[m]	maxPPR[kPa]	maxPFD[m]	maxPFV[m/s]
com1DFAOrig avaAlr	2760.4	5.87	343.6	8.8	41.4
com1DFA avaAlr	2760.7	4.88	342.2	8.6	41.3
com1DFAOrig avaParabola	1982.9	12.4	122.8	3.2	24.7
com1DFA avaParabola	1979.3	12.7	122.8	3.1	24.7

Table 2: Values for avaAlr (real topography) and avaParabola (idealized topography), each computed with com1DFAOrig and com1DFA. Mean, standard deviation, minimum and maximum values for peak values.

Measure	sRunout	maxPPR	maxPFD	maxPFV
min	2752.83	340.2	8.79	41.24
max	2772.61	345.72	8.93	41.58
std	5.87	1.69	0.05	0.1

Table 3: avaAlr (real topography) computed with com1DFAOrig. Standard deviation, Minimum and maximum values for 15 simulations.

Measure	sRunout	maxPPR	maxPFD	maxPFV
min	2752.83	340.03	8.59	41.23
max	2767.67	344.87	8.75	41.53
std	4.88	1.55	0.05	0.09

Table 4: avaAlr (real topography) computed with com1DFA. Standard deviation, Minimum and maximum values for 15 simulations.

Measure	sRunout	maxPPR	maxPFD	maxPFV
min	1959.64	116.18	2.8	24.1
max	1999.64	129.15	3.65	25.41
std	12.49	4.14	0.29	0.42

Table 5: avaParabola results for avaParabola (idealized topography) computed with com1DFAOrig. Standard deviation, Minimum and maximum values for 15 simulations.

Measure	sRunout	maxPPR	maxPFD	maxPFV
min	1954.64	115.87	2.78	24.07
max	1999.64	133.19	3.38	25.81
std	12.74	4.69	0.19	0.47

Table 6: avaParabola (idealized topography) computed with com1DFA. Standard deviation, Minimum and maximum values for 15 simulations.

As a next step, we show the differences introduced when comparing the results of com1DFAOrig and com1DFA. Compared to the 'standardTests' report, where the initial particle location is read from com1DFAOrig for both computational modules, here com1DFA's own initialisation is used. In

figs. 23 and 24, in the left panel, the peak pressure field of the reference simulation is shown. In the right upper panel, a zoom into the runout area including contour lines for different peak pressure thresholds is shown. This provides more insight into the spatial variation of differences between the two simulation results.

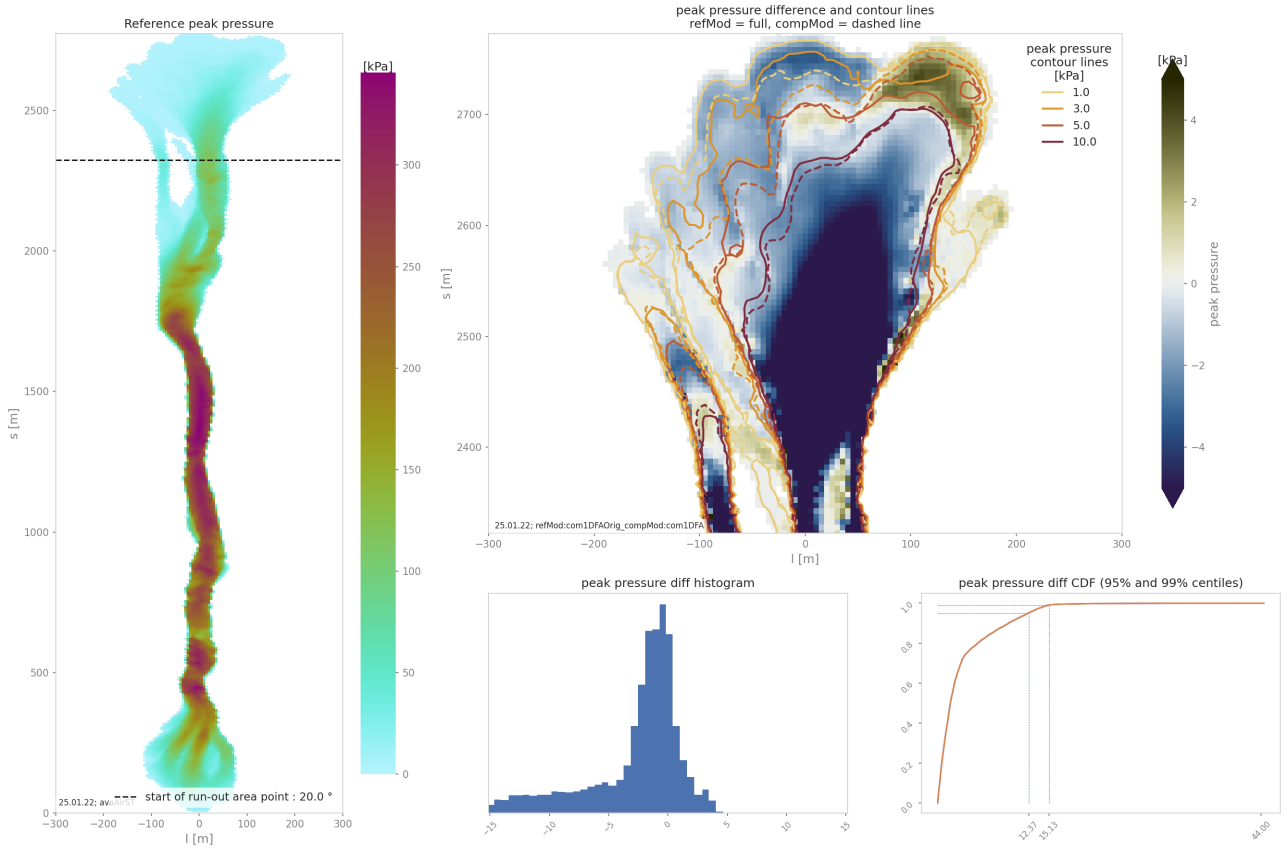

Figure 23: Difference plots of avaAlr (real topography) peak pressure for com1DFAOrig (reference) vs com1DFA (simulation). The left panel shows the peak pressure field in the path following coordinate system for the reference (com1DFAOrig) result. The upper right panels shows the difference (simulation - reference) and contour lines of peak pressure in solid (reference) and dashed (simulation) lines. The lower right panels show a histogram and a cummulative density plot for the differences within the runout area.

In figs. 25 and 26, the mean and maximum values of the crossprofiles along the avalanche path are compared to each other.

One note on these direct comparisons, as shown in the previous probability map figures, choosing a different seed introduces a certain variability into the simulation results. Hence, the magnitude of differences between two simulations strongly depends on the two simulations chosen. Therefore, figs. 25 and 26 show just one example of potential differences.

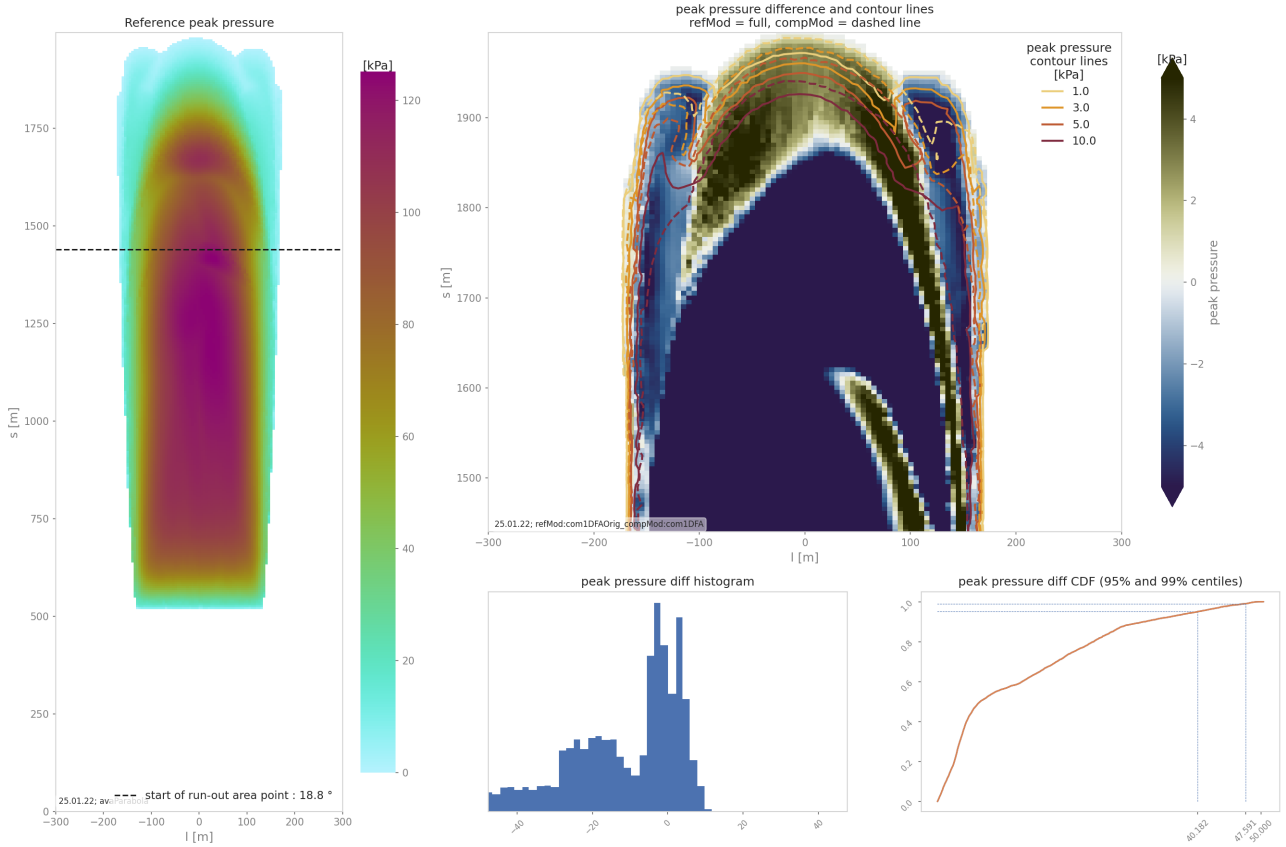

Figure 24: Difference plots of avaParabola (idealized topography) peak pressure for com1DFAOrig (reference) vs com1DFA (simulation). The left panel shows the peak pressure field in the path following coordinate system for the reference (com1DFAOrig) result. The upper right panels shows the difference (simulation - reference) and contour lines of peak pressure in solid (reference) and dashed (simulation) lines. The lower right panels show a histogram and a cummulative density plot for the differences within the runout area.

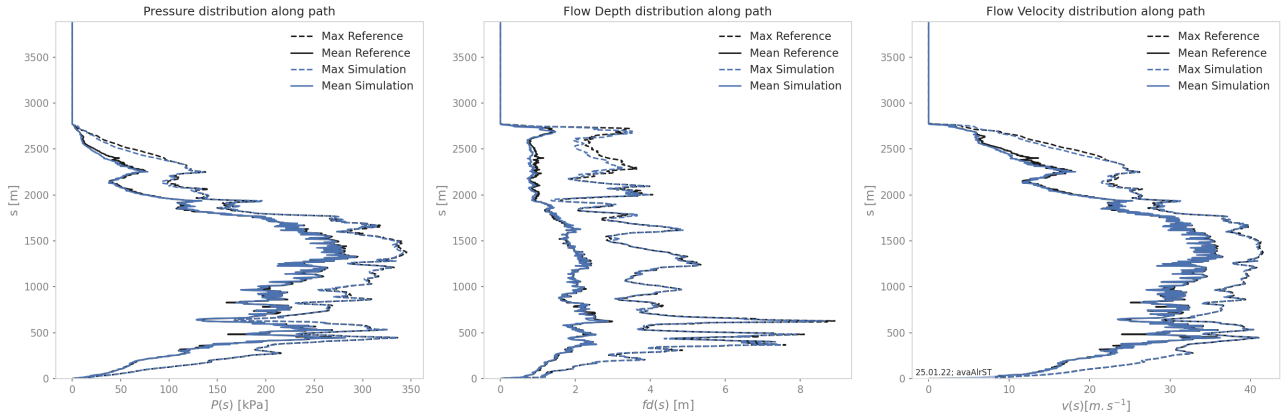

Figure 25: Difference plots of avaAlr (real topography) for com1DFAOrig (reference - in black) vs com1DFA (simulation- in blue), showing the cross-profile mean (solid line) and max (dashed line) values along the avalanche path for peak pressure (left), peak flow depth (middle) and peak velocity (right).

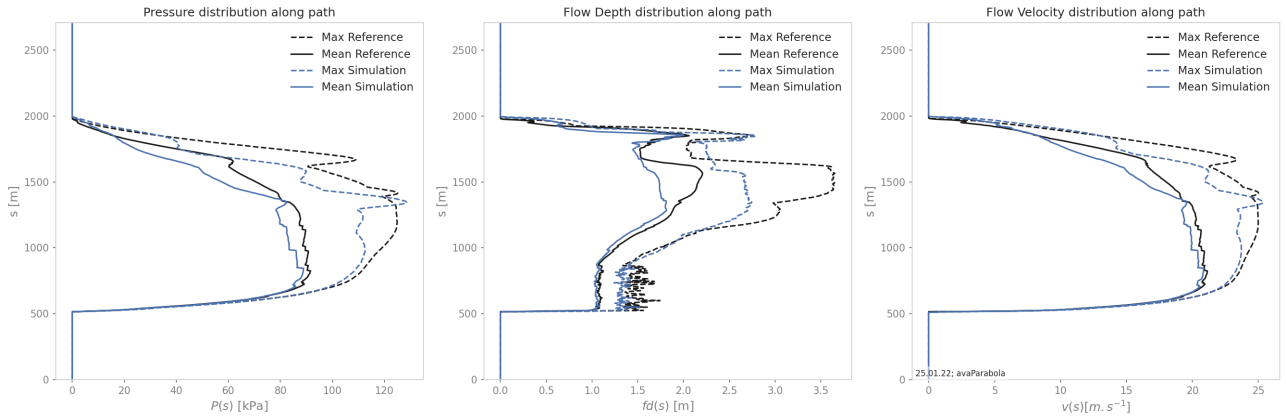

Figure 26: Difference plots of avaParabola (idealized topography) for com1DFAOrig (reference - in black) vs com1DFA (simulation- in blue), showing the cross-profile mean (solid line) and max (dashed line) values along the avalanche path for peak pressure (left), peak flow depth (middle) and peak velocity (right).

8 Outlook for AvaFrame development

Looking at the error ranges displayed in the experiments of this report, we see that random changes of a numerical input parameter (section 7.3; random number seed) lead to differences in results similar to the differences stemming from the different implementations (i.e. com1DFAOrig versus com1DFA). Since the goal of this report is not to investigate the root for the conceptual / methodical differences, but to discern whether com1DFA is close enough to com1DFAOrig, we do not suggest any improvements on the conceptual / methodical level here. We accept the differences as being in a reasonable range and conclude com1DFA producing similar enough results to com1DFAOrig, therefore all future development will use com1DFA as base reference.