Solver updates and new add-ons

`darcyInterTransportFoam` (Pardo-Álvarez, 2025a) widens the applicability of `hyporheicScalarInterFoam` (Lee et al., 2021) through new simulation features. These include not only internal solver updates but also the implementation of external applications, such as several pre- and post-processing utilities or three new boundary conditions. These further developments allow the application of the model to actual, SW-GW contexts at different scales, whose characteristics could not be accounted for with the previous version of the solver.

A detailed description of the solver updates and add-ons of `darcyInterTransportFoam` is provided in the following sections.

S1 Code modifications

Numerous changes were implemented in the code of the model as well as in the structure and organization of both files and directories. Such modifications aim not only to simplify the code but also to make it more accessible for other users. The applied actions are enumerated as follows:

1. Code update to the latest **OpenFOAM** version, v2412, released in December 2024.
2. File reordering and simplification. Only the model-specific files are now contained in the solver directory. The rest are added through the Make/options file, where the paths to the standard-released solver directories are included within the applications section (`EXE_INC`).
3. Removal of unnecessary and unused files and pieces of code (i.e., variables, fields and instructions), as well as code reordering to improve readability.
4. Renaming of files, fields and variables to make them simpler and more intuitive for the user.
5. Definition of the necessary variables, fields and dictionaries for the newly implemented modelling options (described below).

S2 Solver updates

The simulation capabilities of `hyporheicScalarInterFoam` are upgraded in `darcyInterTransportFoam` through new modelling features that either improve previously existing functionalities or provide additional simulation options. The complete description of each model novelty is provided below, including the names of all the cited parameters (i.e., files, variables, fields, dictionaries, etc.) to facilitate its search in the code:

1. As opposed to the original model, the subsurface flow solution is computed based on the hydraulic head field (`hSub`) rather than the modified pressure field (`pTildeSub`). This fact not only makes the code more intuitive for any user but also facilitates its coupling to any other groundwater model. Furthermore, the model calculates the piezometric level and hydraulic head fields (`piezoSur`, `hSur`) in the surface domain, the specific discharge field (`q`) in the subsurface domain and the water depth fields (`dSur`, `dSub`) in both domains.

2. The `dSur`, `piezoSur` and `hSur` fields can be optionally clipped every timestep (`alphaAirClip`) according to an α_i threshold (`alphaWaterThres`) defined in the *transportProperties* dictionary of the surface domain. As a result of the clipping, the three hydraulic variables turn zero at those cells where α_i is below the specified threshold. This action facilities the visualization of the computed water phase as well as its exportation to another flow model.

3. Analogously to other multiphase **OpenFOAM** solvers, **`darcyInterTransportFoam`** allows to define a free-surface reference (`hRef`) in the surface domain. This feature was unable in the original model, which prevented the user to set a (subcritical) water height at the surface outlet patch. Such height is commonly combined with a Dirichlet hydrostatic pressure BC and a von Neumann BC for velocity to model the free exist of the flow from the simulation domain. This approach not only avoids potential stability issues but also prevents unrealistic hydrodynamic scenarios, such as the decrease of the free-surface towards the outlet of the surface domain (supercritical flow).

4. The adjustable body force, $gradP_i$ (`gradP`), required to make $p_{sur,i}$ compatible with periodic boundary conditions (referred to as `cyclic` in **OpenFOAM**) is now considered in the computation of all the subsurface hydraulic variables. $gradP_i$ enables to adjust the surface volume-averaged streamwise velocity field of the momentum equation, $U_{sur,i}$ (`magUbarAve`), to a user-defined mean flow velocity, $\overline{U}$ (`Ubar`), equivalent to an enforcing discharge. Its value is calculated through an external function object (**`VoFmeanVelocityForce`**) at every timestep as:

$$gradP_i = \lambda \frac{(|\overline{U}| - |U_{sur,i}|)}{a_{p_i}}, \tag{S1}$$

where λ (`relaxation`) is a relaxation factor and a_{p_i} (`rAUave`) are the diagonal elements of the linear system of equations of $U_{sur,i}$. In the original model, **`hyporheicScalarInterFoam`**, the pressure gradient $gradP_i$ was prescribed to Darcy's law (`q`) when required by periodicity but left it aside in the hydraulic head (`hSub`) and groundwater flux (`phi_q`) calculation. **`darcyInterTransportFoam`** fills this computational gap by additionally adjusting `hSub` and `phi_q` via the velocity flux resulting from $gradP_i$.

5. The original model defined the hydraulic conductivity (`K`), the porosity (`theta`) as well as the longitudinal and transverse dispersivity (`alpha_L` and `alpha_T`) fields through single values specified in the *transportProperties* dictionary of the subsurface domain. Furthermore, the model did not require a `K` value but calculated it via the intrinsic permeability of the porous medium (`kSub`) and the dynamic viscosity of the water (`muSub`). Consequently,

all the abovementioned subsurface fields could be simply defined as homogeneous and isotropic. This fact prevented the user to, e.g., distinguishing different `theta` and `K` zones in the aquifer or from establishing a heterogeneous distribution of both parameters throughout the domain. In **darcyInterTransportFoam**, however, `K` is directly specified as a tensor field and `theta`, `alpha_L` and `alpha_T` as scalar fields. This new setup provides full flexibility for the distribution of the previous parameters in the subsurface domain (beyond the impervious zone derived from the surface dry areas – see next point). The definition of the non-uniform fields can be performed following different approaches. These include, e.g., the field data assignment before the start of the simulation or the use of *setFieldsDict* through its multiple geometric selection options.

6. **hyporheicScalarInterFoam** could not handle situations where areas of the surface bottom patch were partially dry. This circumstance no longer represents a limitation in **darcyInterTransportFoam**. A newly implemented approach allows the automatic specification of impermeable regions in the subsurface domain according to the surface dry areas. This novel methodology comprises two steps: first, the dry faces at the surface bottom patch are selected (`dryBottomSur`) each timestep based on `alphaWaterThres` (*transportProperties* dictionary of the surface domain). Next, the selected faces are optionally used (`imperRegion`) to define impervious regions in the subsurface domain, whose shape and extension correspond to the surface areas previously chosen. The depth (`d_imper`, must be always negative), the geological properties (`K_imper`, `theta_imper`, `alpha_L_imper` and `alpha_T_imper`) as well as the solute concentration (`CSub_imper`) and water temperature (`TSub_imper`) of such regions are supplied within the *imperRegionProperties* dictionary (*constant* directory of the subsurface domain). Moreover, the user can optionally set a time limit for the update of the specified impermeable zones via `endTimeUpdateRegion` in the same dictionary. The use of this new functionality is limited to quasi-steady-state scenarios in which the dry areas of the surface bottom patch remain constant throughout the entire simulation. Applying it to transient scenarios would result in unrealistic behaviour, as steep hydraulic gradients would be generated at the newly permeable cells each timestep. As an alternative, the user can manually define as many steady, impermeable regions as desired taking advantage of the model novelty number 5.
7. The new model modifies the process selection scheme implemented in **hyporheicScalarInterFoam**, which required to first choose the domain/s to be simulated (i.e., surface and/or subsurface) and next the process/es to be solved (i.e., flow, solute transport and/or heat transfer). Additionally, the solute transport simulation had to be separately selected for each domain. In **darcyInterTransportFoam**, the choice of the flow, the solute transport as well as the heat transfer simulation can be made jointly for both domains in the *simulationProperties* dictionary (`flow`, `soluteTransport` and `heatTransfer` options, respectively). As an alternative to the joint selection, the simulation of each process can also be specified for a single domain in its corresponding *transportProperties* dictionary (*constant* directory). In this latter case, the simulation options would be `flowSur`, `soluteTransportSur`

and `heatTransferSur` for the surface domain and `flowSub`, `soluteTransportSub` and `heatTransferSub` for the subsurface domain.

8. The solute transport module has been completely reformulated in the new model. The implemented changes affect both the code organization and the modelling approach. Regarding the former, the diffusion-dispersion coefficient expressions as well as the solute transport governing equations are now placed within one single header file (*.H) in each domain. These new *.H files for the surface and subsurface domains are respectively named as *CSurEqn.H* and *CSubEqn.H*. As for the computational approach, both surface diffusion and subsurface hydrodynamic dispersion fields (`DSur` and `DSub`, respectively) are calculated in a different way than in **hyporheicScalarInterFoam**. `DSur` is computed through a new expression, whereas `DSub` results from a partial modification of the original equation. Additionally, both `DSur` and `DSub` can be alternatively specified as uniform and constant over the entire domain through a single value (`constDSur` or `constDSub`). Furthermore, **darcyInterTransportFoam** provides a new simulation option to solve the conservative solute transport (`CSur`) in just one phase at the surface domain (`solutePhases == "phase1"`). This novel feature is based on the advection-diffusion equation contained in the standard **scalarTransport** function object of **OpenFOAM**. Yet, the two-phase solute simulation remains as an option through the original computational approach (`solutePhases == "all"`). All the modelling options and parameters required to conduct the solute transport simulation have to be specified in the *transportProperties* dictionary of each domain.
9. The simulation of the temperature distribution in both domains is now available thanks to the heat transfer module implemented in **darcyInterTransportFoam**. The code organization as well as the computational scheme of this new module are very similar to those applied to the solute transport module. In this way, all the equations required to conduct the heat transfer simulation are included within the *TSurEqn.H* and *TSubEqn.H* files, each belonging to one domain. The latter contains the partial differential equation required to compute the temperature distribution in the ground (`TSub`), whereas the former comprises the same phase-based simulation options (`heatPhases`) available for solute transport to solve the transfer of heat in the surface domain (`TSur`). The necessary parameters and simulation options to run the new temperature module need to be defined in the *transportProperties* dictionary corresponding to each domain.
10. The sequential iterative coupling scheme has been modified in the new model to improve the transfer of information between the domains. In this way, the data is no longer interpolated but directly assigned to the matching nodes of both meshes at their respective interface patch. This new approach guarantees the exact conveyance of the solution from one domain to another (`hSur`, `q`, `CSur/CSub` and `TSur/TSub`), avoiding the potential loss of information resulting from the former path-to-patch interpolation. As a result, the two-way data mapping is performed in a more accurate way than in the original model, **hyporheicScalarInterFoam**.

11. The convergence check of both domain solutions is reformulated in the new model. Flow and solute convergences are now evaluated every timestep by means of flux mismatch at the interface patch of the two meshes, that is, the surface bottom and subsurface top boundaries. The solution convergence is reached once both water and conservative solute flux mismatches (`couplePhiMaxMismatch` and `coupleSolutePhiMaxMismatch`, respectively) are below the specified tolerances (`couplePhiTol` and `couplePhiSoluteTol`, respectively) in the *simulationProperties* dictionary. Further work will be still required in the future to improve the implemented convergence scheme.

12. To make the most of the existing coupling modes to run the model (one and two-way), a single-domain simulation feature was implemented in **`darcyInterTransportFoam`**. The new add-on enables to define different boundary conditions (BCs) at the interface patch of each domain when the latter are run alone. In this way, both the surface velocity (`U`) and the subsurface hydraulic head (`hSub`) fields can be either specified as the solution from a previous fully-coupled simulation (e.g., `interUSur == "latestTime"`) or set as homogeneous (e.g., `interhSub == "consth"`) via a user-defined single value (`Ubottom` and `hSubTop`, respectively) at the corresponding interface boundary. In addition to the aforementioned, the solute concentration (`CSur` and `CSub`) and the water temperature (`TSur` and `TSub`) fields also allow to set a `zeroGradient` condition at the surface bottom and subsurface top patches. The latter is defined as the default BC for both `CSu*` and `Tsu*` fields, in contrast to the `latestTime` option used by `U` and `hSub`. Both the BC choice (e.g., `interTSur = "zeroGradient"` or `interCSub = "constC"`) and its value (in case the homogeneous option is selected) need to be specified in the *transportProperties* dictionary of the corresponding domain. This novel model feature enables alternative switching between coupling modes within the same simulation in order to take full advantage of the benefits of each to optimize the computation.

13. **`darcyInterTransportFoam`** allows to reinitiate the data fields at any time of the simulation skipping the time-consuming data-preprocessing procedure. This can be done thanks to the multiple, newly implemented reset options. To perform the reset, the new model requires from at least three data: (1) the initial time of the new simulation (`initialTime`), (2) the simulated time from which the reset values are taken (`resetTime`) and (3) the reset option that indicates the data to be restarted (e.g., `resetSur` or `resetHeatSub`). The definitions of both `initialTime` and `resetTime` are optional. By default, the model assigns the latest simulated time to `initialTime` and the initial simulated time to `resetTime`. Additionally, when `initialTime` is not defined by default, the timesteps between `initialTime` and the latest simulated time can optionally be removed via `rmInterTimes`. Regarding the choice of the data fields to be reset, the `resetSur` and `resetSub` options allow to restart, respectively, all the surface and subsurface fields. As an alternative to these, the reset can be limited to the solute transport and heat transfer fields through the `resetSoluteSur/resetSoluteSub` options for the surface domain and the `resetHeatSur/resetHeatSub` options for the subsurface domain. Both reset times and data field options have to be specified in the *resetOptions* dictionary located in the *constant* directory.

S3 Utilities

Four new utilities are available to be used in combination with the new fully-coupled model. Some of them, additionally, are not exclusive of **`darcyInterTransportFoam`** but also allow its application with other **OpenFOAM** solvers (e.g., **`interFoam`** or **`pisoFoam`**). An extended description of each novel utility is provided below:

- **`checkMeshesMatch`**: checks whether the number of faces of both meshes and their coordinates match at the interface patch (`interfacePatch`). The coordinate check is done via a user-specified distance threshold (`distThres`) after the face number condition is verified. Failure to comply with any of the previous requirements will immediately stop the utility execution and provide an error message. The fulfilment of both conditions guarantees the appropriate fitting of the surface and subsurface meshes. This utility is model-specific, meaning it can only be applied with **OpenFOAM** models consisting of two adjacent simulation meshes, such as **`darcyInterTransportFoam`**.
- **`changeZRefMesh`**: optionally (1) translates the mesh points (`transformZmesh`) according to a user-specified reference mesh elevation (`zRef`) and (2) adjusts the `hRef` value to the actual mesh location (`waterHeightOutlet`). These two actions are independent; the former does not depend on the latter, and vice versa. Whereas the first option allows to move the mesh to the location specified by `zRef`, the second transforms the user-defined water depth (i.e., `hRef`) into water height by adding the minimum mesh elevation at the outlet to `hRef`. This utility can be applied to one (surface) or two (surface and subsurface) domains (`numDomains`), being thus possible its use beyond **`darcyInterTransportFoam`**.
- **`restoreZRefMesh`**: optionally (1, 2) restores the two actions performed by **`changeZRefMesh`** (i.e., `transformZmesh` and `waterHeightOutlet`) and (3) additionally adjusts the computed hydrological fields `piezoSur`, `hSur` and `hSub` to the final mesh location (`updateHydroFields`). All these actions are independent, meaning that none of them requires from another to be performed. This utility translates the mesh points (*polyMesh* directory) based on the `meshShiftZ` value, which is either previously determined by **`changeZRefMesh`** or supplied by the user in the *constant* directory. The latter option will be always required if **`changeZRefMesh`** is not executed before **`restoreZRefMesh`**. `meshShiftZ` is additionally used to update the `piezoSur`, `hSur` and `hSub` fields according to the resulting mesh elevation. Moreover, `hRef` is reestablished to its original, user-defined value by subtracting the minimum mesh elevation at the outlet to the formerly calculated water height. Analogously to **`changeZRefMesh`**, this utility is also applicable to one (surface) or two (surface and subsurface) domains (`numDomains`). Consequently, its use is not limited to the new fully-coupled model but also open to other **OpenFOAM** solvers. The utility setup and its outcomes will ultimately depend on the prior application of **`changeZRefMesh`**.

- **`transferMeshDecomp`**: transfers the domain decomposition from one mesh to another through the interface patch of both meshes (`interfacePatch`). Moreover, it optionally overwrites the *cellDecomposition* file contained in the *constant* directory of each domain. This utility requires a previous decomposition of each domain (**`decomposePar`** utility) that generates the necessary *cellDist* and *cellDecomposition* files for its execution. The decomposition transfer is then conducted by subdomain in two steps (`nSubDomains`). First, the subdomain extension is delimited at the interface path of the reference mesh (`rbDeltaX`, `rbDeltaY` and `rbDeltaZ`). Next, the corresponding subdomain ID is assigned to all the cells of the target mesh falling inside the previously defined limits (`distThres`). This procedure is possible in two ways, that is, from the surface to the subsurface and vice versa (`transferDir`). Once the transfer is done, the user can decide whether or not overwriting the *cellDecomposition* file of the reference (`overWriteCellDecompBase`) and/or target mesh (`overWriteCellDecomp`). In the same way as **`checkMeshesMatch`**, the application of this utility is not strictly limited to **`darcyInterTransportFoam`** but to **OpenFOAM** models comprised by two adjacent meshes.

S4 Boundary conditions

Three new boundary conditions, one for the surface velocity field, $U_{sur,i}$ (m/s), and two for the subsurface hydraulic head field, $h_{sub,i}$ (m), were developed to be used with **`darcyInterTransportFoam`**. The groundwater BCs set implicitly the heads, since their value is either retrieved from the surface domain or inferred from the prescribed velocity at the patch. Moreover, as happens with the aforementioned utilities, some of the novel BCs do not restrict its application to **`darcyInterTransportFoam`** but also allow its use with other standard **OpenFOAM** solvers. A detailed description of all the new BCs can be found within the following points:

- **`flowRateInletParabolicVelocity`**: optionally prescribes at the selected patch an inward-pointing normal, parabolic $U_{sur,i}$ (`U`) based on the specified flow rate, Q_{sur} (m^3/s - `volumetricFlowRate`), the dimensions of the velocity profile (`Uprofile == "parabolic"` - 2D - or `"paraboloidal"` - 3D), the type of inlet geometry (`geometryInlet == "open"` or `"closed"`), the average velocity multiplier, $multi_{\bar{U}}$ (`avgUmulti`), and the water depth direction (`depthDir`). The shape of the resulting velocity profile (either a half or a quarter of a parabola) will primarily depend on the geometry type of the inlet (Fig. S1). This new BC allows to avoid potential numerical issues at the conjunction with no-slip patches (commonly bottom and side boundaries) when applying a uniform velocity field. The boundary condition setup results from the combination of the standard **`flowRateInletVelocity`** BC and the **`parabolicVelocity`** BC developed by Hrvoje Jasak, freely available in **foam-extend-5.0**. As an example of its functionality, the following equation shows how the paraboloidal velocity profile is determined at a closed inlet patch:

$$U_{sur,i} = \vec{n}_i \cdot multi_{\bar{U}} \cdot \frac{Q_{sur}}{A_{p_{w,i}}} \cdot \left(1 - \sqrt{\frac{(\vec{f}_{c_i} - \vec{p}_c) \cdot \overrightarrow{dir}_H}{0.5 \cdot H}}\right) \cdot \left(1 - \sqrt{\frac{(\vec{f}_{c_i} - \vec{p}_c) \cdot \overrightarrow{dir}_W}{0.5 \cdot W}}\right), \tag{S2}$$

where n_i is the patch face unit normal vector field (-), $A_{p_{w,i}}$ is the area of the patch faces through which water flows into the domain (m^2), $\vec{f}_{c_i}$ is the patch face centre vector field (m), $\vec{p}_c$ is the paraboloid centre vector (m), $\overrightarrow{dir}_H$ and $\overrightarrow{dir}_W$ are respectively the patch height and width direction vectors (-) and H and W are the height and width of the patch (m). Analogously to most standard velocity BCs, the use of **`flowRateInletParabolicVelocity`** is not limited to the new model but extensible to other **OpenFOAM** models.

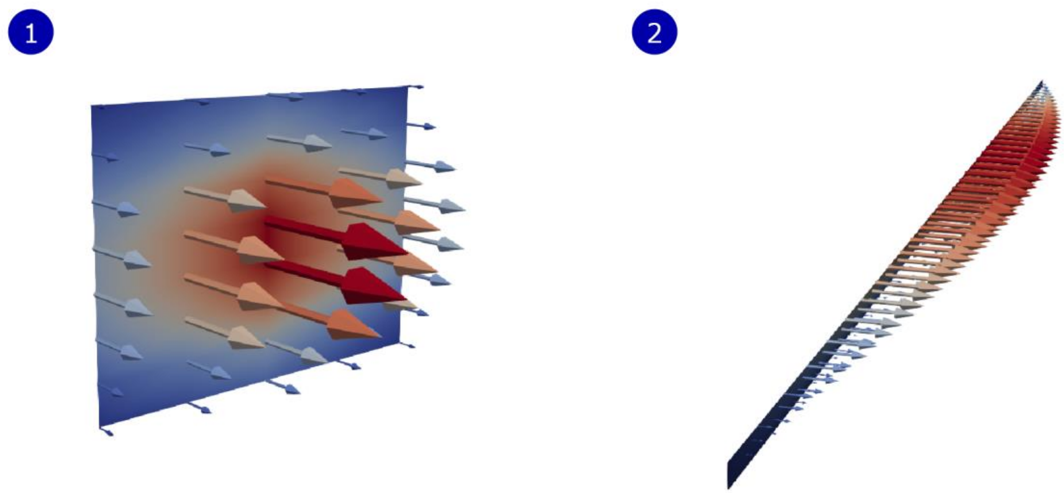

Figure S1: `flowRateInletParabolicVelocity` BC paraboloidal velocity profile: (1) half paraboloid generated at a closed inlet patch and (2) quarter of a paraboloid prescribed at an open inlet patch.

- **`darcyGradHead`**: sets at the patch the $\nabla h_{sub,i}$ resulting from the flux generated by the specific discharge, q_i (m/s), boundary condition. Such $h_{sub,i}$ - q_i (`hSub - q`) relationship is inferred from Darcy's law as:

$$\vec{n}_i \cdot \nabla h_{sub,i} = -\frac{1}{K_{ij}} \cdot \vec{n}_i \cdot q_i, \tag{S3}$$

where K_{ij} is the hydraulic conductivity tensor field (m/s). This new BC provides an alternative way to specify $h_{sub,i}$ for cases where the definition of q_i is more convenient than the direct assignment of $h_{sub,i}$ at the patch. Originally inspired by the standard **`fixedFluxPressure`** BC, **`darcyGradHead`** has been adapted from **`darcyGradPressure`**, a head BC previously developed by Cyprien Soulaine as part of the **`porousMultiphaseFoam`** toolbox (Horgue et al., 2015). Similarly to **`flowRateInletParabolicVelocity`**, its application to **OpenFOAM** models other than **`darcyInterTransportFoam`** is also possible.

- **`surfaceHydrostaticHead`**: specifies at the inlet, outlet or aquifer (i.e., bottom) patch of the subsurface domain a hydrostatic head BC based on the $h_{sub,i}$ values obtained in the surface domain. In this way, if applied to the subsurface inlet or outlet, this BC will take and assign the hydrostatic head value, $max(piezo_{sur,i})$, computed at the equivalent patch of the surface domain. Analogously, if set at the subsurface bottom boundary, it will retrieve the hydrostatic head field, $piezo_{sur,i}$, from its surface counterpart. The current version of **`surfaceHydrostaticHead`** does not allow its application to any patches beyond the cited. This novel boundary condition avoids the unrealistic behaviour caused by the commonly used **`zeroGradient`** BC. The latter works as an impermeable layer, forcing the flow paths to change their direction in the vicinity of the patch. **`surfaceHydrostaticHead`**, however, provides a closer approximation to reality and allows the water to flow across the boundary. The direction of the fluxes will depend on the sign of the $\nabla h_{sub,i}$ at the patch. Moreover, in contrast to the other two new BCs, the present one is model-specific, meaning it can only be used with **`darcyInterTransportFoam`**.

References

Horgue, P., Soulaine, C., Franc, J., Guibert, R., and Debenest, G.: An open-source toolbox for multiphase flow in porous media, Comput. Phys. Commun., 187, 217–226, https://doi.org/10.1016/j.cpc.2014.10.005, 2015.

Lee, A., Aubeneau, A., Liu, X., and Cardenas, M. B.: Hyporheic exchange in sand dunes under a freely deforming river water surface, Water Resour. Res., 57, e2020WR028817, https://doi.org/10.1029/2020WR028817, 2021.

Pardo-Álvarez, Á.: darcyInterTransportFoam v1.0, Zenodo [code], https://doi.org/10.5281/zenodo.15857142, 2025a.