The Journal of Open Source Software

# dorado: A Python package for simulating passive particle transport in shallow-water flows

**Jayaram Hariharan**[1], **Kyle Wright**[1], **and Paola Passalacqua**[1]

**1** Department of Civil, Architectural and Environmental Engineering, The University of Texas at Austin

**DOI:** 10.21105/joss.02585

**Software**

- Review
- Repository
- Archive

**Editor:** Katy Barnhart

**Reviewers:**

- @dbuscombe-usgs
- @gassmoeller

**Submitted:** 11 August 2020
**Published:** 22 October 2020



## Summary

Hydrodynamic simulations of flow through landscapes allow scientists to answer questions related to the transport of water, nutrients, pollutants, biota, and sediment through waterways (Czuba, David, Edmonds, & Ward, 2019; Duan & Nanda, 2006; Lauzon & Murray, 2018; Rynne, Reniers, Kreeke, & MacMahan, 2016; Wild-Allen & Andrewartha, 2016). In geophysical systems, such as rivers, estuaries, and deltas, hydrodynamic models typically solve the depth-integrated "shallow water" equations in an Eulerian reference frame, which is concerned with fluxes through a given region of space – examples of these solvers include ANUGA ("ANUGA," 2019), Delft3D ("Delft3D," 2020), Frehd (Hodges, 2014) and others. However, the spatial and temporal characteristics of the movement of material through a landscape are often better understood using a Lagrangian reference frame (Doyle & Ensign, 2009), which follows the movement of individual objects or parcels. In this paper, we present an open-source Python package, *dorado*, which provides a transparent and accessible method for researchers to simulate passive Lagrangian particle transport on top of Eulerian hydrodynamic solutions. This mixed Eulerian-Lagrangian methodology adapts the routing functionality from the popular numerical model DeltaRCM (Liang et al., 2015a, 2015b) for use with the outputs of any shallow-water hydrodynamic solver.

## Statement of Need

Existing software for tracking water parcels or particles in a Lagrangian framework typically contains complicated code structures ("Delft3D," 2020; Froger & Souille, 2017) with steep learning-curves, are proprietary ("ANSYS Fluent Software: CFD Simulation," 2020), or were developed to solve problems at very different spatial scales (Dagestad, Röhrs, Breivik, & Ådlandsvik, 2018; Yeung & Pope, 1989) than typical riverine applications. This package fills a gap in the available methods for Lagrangian particle simulation by providing a flexible, open, and transparent framework meant for rapid application in conjunction with any landscape-scale 2D hydrodynamic model. In addition, *dorado* comes with built-in pre-processing, analysis, and plotting functionality, including methods to compute the exposure time distribution of particles to specific sub-regions of the domain, which is often of interest in ecological applications (Kadlec & Wallace, 2008).

## Background

*dorado* makes use of the weighted random walk framework (Pearson, 1905) to model particle transport as a stochastic Markovian process, using a grid-centric approach that takes into

account local water inertial components and surface slopes, in a manner modeled after the numerical model DeltaRCM (Liang et al., 2015a, 2015b). The random walk routing weights are sensitive to two user-specified parameters, $\gamma$ and $\theta$. The $\gamma$ parameter controls the proportional importance of water surface gradient and water velocity in determining the downstream routing direction $F^*$, according to:

$$F^* = \gamma F_{sfc} + (1 - \gamma) F_{int} \tag{1}$$

in which $F_{sfc}$ and $F_{int}$ represent the surface gradient and inertial components, respectively. Modifying $\gamma$ indirectly controls the diffusivity of the travel path. The second weighting parameter, $\theta$, modifies the routing weights, $w_i$, for the random walk based on the local water depth value, $h_i$. The routing weights for each neighboring cell are calculated per Liang et al. (2015b):

$$w_i = \frac{h_i^{\theta} \max\left(0, F \cdot d_i\right)}{\Delta_i} \tag{2}$$

wherein $F$ are the routing directions, $d_i$ represents the unit vector from the current cell to its neighbor, and $\Delta_i$ is the D8 cellular distance. Here, the local flow depth acts as a resistance term – as $\theta$ gets larger, the local routing weights have an increasingly large dependence on the water depth of the neighboring cells. By altering the two parameters $\gamma$ and $\theta$, the user has control over the randomness with which the particles are routed. For a full description of this methodology, see Liang et al. (2015a, 2015b) and the *dorado* documentation.

Particle travel times are back-calculated from their locations after routing by accounting for the flow velocity, flow direction, and grid size. The time elapsed during each particle step is assumed to be proportional to the distance traveled in the direction of the mean flow, $d_{eq}$, which is equal to the step distance $\Delta_i$ projected onto a unit vector oriented in the direction of the velocity field, $\phi$, according to $d_{eq} = \Delta_i \cdot \cos(\phi)$. During a model iteration, each particle is allowed to progress a different length of time, depending on the velocity of the flow in the vicinity of each particle and the orientation of each step in relation to the mean flow direction. The travel time of each step is then modified according to a user-specified parameter, $D_c$, which acts as a dispersion coefficient.

*dorado* provides functions with which users can choose to sync up the particle evolution by either number of step iterations or by individual particle travel times. In addition, *dorado* includes several post-processing functions for computing and plotting the fraction of time particles spend "exposed" to a region of interest, otherwise known as the exposure time distribution (or residence time distribution, in steady flows, Kadlec & Wallace, 2008; Benjamin & Lawler, 2013; Hiatt, Castañeda-Moya, Twilley, Hodges, & Passalacqua, 2018). This travel time methodology has been tested against analytical solutions for a plug-flow reactor with dispersion (Benjamin & Lawler, 2013), as well as against exposure time distributions from prior models of real systems (Hiatt et al., 2018), and performed well at reproducing the observed travel times in both systems. As with all Lagrangian methods involving finite samples of particles, we expect the travel time computations to struggle to reproduce the heavy-tailed behavior observed in some systems (Zhang, Meerschaert, & Packman, 2012), and expect that *dorado* will perform most accurately in advection-dominated flows.

## Functionality and Ease of Use

*dorado* is capable of handling both steady and unsteady hydrodynamic simulations. While the core functionality of *dorado* assumes that flow variables lie on a Cartesian grid of uniform grid size, the package can still be applied to unstructured hydrodynamic models through the use of built-in interpolation functions for gridding data. Due to the nature of the weighting scheme, model flow-fields must represent depth-averaged solutions (as vertical movement of particles is not considered). *dorado* supports current Python 3.x versions as well as the older

2.7 version, can be run on all major operating systems, and is dependent upon only a select few common scientific packages. To help new users, over a dozen examples are provided, illustrating uses of the package with DeltaRCM and ANUGA model simulations conducted on real and synthetic datasets. The package is organized into four modules to help users and future contributors easily access and interact with the source code:

- `lagrangian_walker`: Contains internal functions and methods associated with the weighted random walk.
- `particle_track`: Lower-level functionality in which the `modelParams` and `Particles` classes are defined, and parameters for the domain and particle evolution are set.
- `routines`: Higher-level functions for common applications, designed to simplify use of the package.
- `parallel_routing`: Parallel functionality which enables the user to distribute their particle routing commands across multiple CPU cores using Python's built-in multiprocessing library.

## Acknowledgements

We thank Nelson Tull for testing and providing feedback on the package. Thanks also to David Mohrig, Kathleen Wilson, Hima Hassenruck-Gudipati, Teresa Jarriel, Eric Prokocki, John Swartz, Shazzadur Rahman, and Andrew Moodie for their input and feedback during development. We also thank Mickey Lanning for suggesting the package name.

This work was supported in part by NSF EAR-1719670, the NSF GRFP under grant DGE-1610403, and the NASA Delta-X project, which is funded by the Science Mission Directorate's Earth Science Division through the Earth Venture Suborbital-3 Program NNH17ZDA001N-EVS3.

## References

ANSYS Fluent Software: CFD Simulation. (2020). Retrieved from https://www.ansys.com/products/fluids/ansys-fluent

ANUGA. (2019). Australian National University; Geoscience Australia. Retrieved from https://anuga.anu.edu.au/

Benjamin, M. M., & Lawler, D. F. (2013). *Water quality engineering: Physical/chemical treatment processes*. John Wiley & Sons.

Czuba, J. A., David, S. R., Edmonds, D. A., & Ward, A. S. (2019). Dynamics of surface-water connectivity in a low-gradient meandering river floodplain. *Water Resources Research*, *55*(3), 1849–1870. doi:10.1029/2018WR023527

Dagestad, K. F., Röhrs, J., Breivik, O., & Ådlandsvik, B. (2018). OpenDrift v1.0: A generic framework for trajectory modelling. *Geoscientific Model Development*, *11*(4), 1405–1420. doi:10.5194/gmd-11-1405-2018

Delft3D. (2020). Deltares. Retrieved from https://oss.deltares.nl/web/delft3d/home

Doyle, M. W., & Ensign, S. H. (2009). Alternative reference frames in river system science. *BioScience*, *59*(6), 499–510. doi:10.1525/bio.2009.59.6.8

Duan, J. G., & Nanda, S. K. (2006). Two-dimensional depth-averaged model simulation of suspended sediment concentration distribution in a groyne field. *Journal of Hydrology*, *327*(3-4), 426–437. doi:10.1016/j.jhydrol.2005.11.055

Froger, D., & Souille, F. (2017). Freshkiss3D. ANGE Team. Retrieved from https://freshkiss3d.gforge.inria.fr/

Hiatt, M., Castañeda-Moya, E., Twilley, R., Hodges, B. R., & Passalacqua, P. (2018). Channel-island connectivity affects water exposure time distributions in a coastal river delta. *Water Resources Research*, *54*(3), 2212–2232. doi:10.1002/2017WR021289

Hodges, B. R. (2014). A new approach to the local time stepping problem for scalar transport. *Ocean Modelling*, *77*, 1–19. doi:10.1016/j.ocemod.2014.02.007

Kadlec, R. H., & Wallace, S. (2008). *Treatment wetlands*. Boca Raton, FL: CRC press. doi:10.1201/9781420012514

Lauzon, R., & Murray, A. B. (2018). Comparing the cohesive effects of mud and vegetation on delta evolution. *Geophysical Research Letters*, *45*(19), 10, 437–10, 445. doi:10.1029/2018GL079405

Liang, M., Geleynse, N., Edmonds, D. A., & Passalacqua, P. (2015a). A reduced-complexity model for river delta formation - Part 2: Assessment of the flow routing scheme. *Earth Surface Dynamics*, *3*(1), 87–104. doi:10.5194/esurf-3-87-2015

Liang, M., Voller, V. R., & Paola, C. (2015b). A reduced-complexity model for river delta formation - Part 1: Modeling deltas with channel dynamics. *Earth Surface Dynamics*, *3*(1), 67–86. doi:10.5194/esurf-3-67-2015

Pearson, K. (1905). The problem of the random walk. *Nature*, *72*(1867), 342. doi:10.1038/072342a0

Rynne, P., Reniers, A., Kreeke, J. van de, & MacMahan, J. (2016). The effect of tidal exchange on residence time in a coastal embayment. *Estuarine, Coastal and Shelf Science*, *172*, 108–120. doi:10.1016/j.ecss.2016.02.001

Wild-Allen, K., & Andrewartha, J. (2016). Connectivity between estuaries influences nutrient transport, cycling and water quality. *Marine Chemistry*, *185*, 12–26. doi:10.1016/j.marchem.2016.05.011

Yeung, P.-K., & Pope, S. B. (1989). Lagrangian statistics from direct numerical simulations of isotropic turbulence. *Journal of Fluid Mechanics*, *207*, 531–586. doi:10.1017/S0022112089002697

Zhang, Y., Meerschaert, M. M., & Packman, A. I. (2012). Linking fluvial bed sediment transport across scales. *Geophysical Research Letters*, *39*(20). doi:10.1029/2012GL053476