The California Department of Water Resources maintains Delta Simulation Model 2, a one dimensional hydrodynamic and transport model for rapidly simulating flow and water quality in the Sacramento-San Joaquin Delta. Recently, the authors commenced work on a more flexible and more rigorously verified transport component for this suite. Our target problems include river and estuary advection, 1D approximations of common mixing mechanisms and source terms associated with sediment, radiation and non-conservative water quality kinetics.

In this paper, we describe our approach and experiences developing a software verification framework for this model. We begin by describing the motivation and requirements for such a framework, and associate our criteria to concepts from both the software and numerical testing fields. We then describe the components and implementation of the suite, emphasizing the incremental nature of the tests, quantitative criteria for testing and the tension between the silent, automatic perspective of software testing and the verbose, graphical requirements required for public reporting of numerical results.

Both the scaling of the problem and our choice of algorithm influence the components of our test suite. The dynamics are formulated in conservative form as follows:

Our problem domain includes estuaries and river channels and even some open water areas grossly approximated as channels. The mixing mechanisms we anticipate are \*\*\*\*. We anticipate blah to dominate blah, with Blah Numbers exceeding \*\*\*. We also contemplate significant, non-linear source terms. While none are so quickly varying as to constitute truly stiff reactions, we have \*\*\* with \*\*\*\* Numbers of \*\*\*\*. O

Our choice of algorithms includes \*\*\*\* advection with \*\*\*\* and \*\*\*\*. The advection and reaction solver are coupled as a predictor corrector pair, and diffusion is implemented using operator splitting. Two features of the algorithm are particularly important. First, our solver requires a flow field (\*\*\* and \*\*\*) that preserves mass continuity. In some cases tests from the literature were written in non-conservative or primitive form in terms of a velocity and had to be reworked in conservative form. Secondly, we employ operator splitting and wanted to exercise the equations with and without known vulnerabilities (such as time-varying boundaries and nonlinear source terms) of this class of algorithm.

In general, we sought strict second order accuracy for individual operators and near second-order accuracy for the algorithm as a whole. The classic advantage of second order is efficient: second order allows coarser discretization for a modest increase in work. As computer architectures favor multiple operations with minimal movement of data, this advantage seems to be on the increase. A second order algorithm also gives us a buffer of accuracy as details like networks of channels and coarse boundary data are added.