ERDC/EL TN-23-3
September 2023

Linking the SEDLZJ Portable Standalone Library to the CMS Coastal Hydrodynamic Model

by Terry Gerald

PURPOSE: This document describes the repackaging and linkage of the Sandia National Laboratories Environmental Fluid Dynamics Sediment Processes Code (SNL-EFDC-SEDZLJ), (Thanh et al. 2008). It was originally incorporated within a modified version of the US Environmental Protection Agency's (USEPA) EFDC public-domain surface-water flow, sediment transport, and water-quality model developed by John Hamrick (Hamrick 1992) and its linkage to the ERDC-CHL-CMS hydrodynamic model.

SNL-EFDC simulates flow and transport of sediment as bedload and suspended load. SNL-EFDC-SEDZLJ improves EFDC with updated sediment kinetics subroutines. Sediment erosion is calculated using data collected with a Sediment Erosion at Depth flume (SEDflume). SEDflume measures erosion rates as a function of shear stress and depth from relatively undisturbed cores taken directly from the sediment bed below the water body of interest. The use of SEDflume data provides more accurate sediment erosion rates that are directly input to the model.

INTRODUCTION: SEDZLJ has been recast as a stand-alone portable library (LIBSEDJ) that may be linked to an array of different hydrodynamic models. LIBSEDJ maintains the original FORTRAN source code as developed by Sandia National Labs (SNL) but includes modifications and additional routines to allow a seamless link to C-type languages. This allows the LIBSEDJ sediment transport library to be linked into FORTRAN hydrodynamic models such as EFDC and CMS, as well as C language models such as ADH. Error fixes and process enhancements made to LIBSEDJ as it evolves in time leverages the hydrodynamic models that are explicitly linked to it.

SEDLIBJ DESCRIPTION: SEDLIBJ routines and data structures may be directly used by a host model or indirectly linked through an abstract user-defined-type named "atype_sedzlj" that is included within the SEDLIBJ library to help users interface to SEDLIBJ quickly and efficiently. This "interface" type aggregates LIBSEDJ routines and data into a single type that may be used as a base class for a host-specific derived class. Utilizing this type as a base class organizes the interface or linkage code automatically leading to a clean separation between host and library code. This approach is used for backporting SEDLIBJ to the EFDC model and linking SEDLIBJ to the CMS model as described in this technical note.

This SEDLIBJ interfacing type is defined in the SEDLIBJ module "atype_sedzlj_mod" as listed below:

```
module atype_sedzlj_mod

 use mod_parameters
 use type_db_mod
 use type_dbc_mod
 use type_wave_mod
 use type_column_mod
 use type_capf_mod
 use mod_sandia_io
 use mod_bedload
 use mod_bedmechanics

 implicit none
 private

 type, abstract, public :: atype_sedzlj

 integer :: ncols = 0

 type(type_capf), allocatable, public :: capfs(:)

 type(type_db), pointer, public :: db => null()

 contains

 procedure, public :: do_initialize => base_sedic

 procedure, public :: do_bed_mechanics => base_bed_mechanics

 ! update "this" with host info
 procedure(IUpdate), deferred, pass(this) :: do_update_capfs

 ! update host with "this" info
 procedure(IUpdate), deferred, pass(this) :: do_update_host

 end type atype_sedzlj

 interface ! For module procedures

 module subroutine base_sedic(this, ncols, bed_elev)
 class(atype_sedzlj), target :: this
 integer :: ncols
 real(kind=zpp) :: bed_elev(:)
 end subroutine base_sedic

 module subroutine base_bed_mechanics(this, delt, ncols)
 class(atype_sedzlj), target :: this
 real(kind=zpp) :: delt
 integer :: ncols
 end subroutine base_bed_mechanics

 end interface

 abstract interface ! For deferred procedures

 subroutine IUpdate(this, ncols)
 import atype_sedzlj
 class(atype_sedzlj), target :: this
```

```
 integer :: ncols
 end subroutine IUpdate

 end interface

end module atype_sedzlj_mod
```

Note that "atype_sedzlj" is an abstract type and cannot be instantiated directly–it can only be implicitly created by instantiating another type that inherits directly from it.

ATYPE_SEDZLJ DATA DESCRIPTIONS: The "atype_sedzlj" class defines three data objects:

1. *ncols*: The total number of sediment columns in the host/SEDZLJ sediment domain.

2. *capfs*: An array of SEDLIBJ derived types each of which abstracts a SEDZLJ sediment column of data and procedures.

3. *db*: A SEDLIBJ derived type that encapsulates all SEDZLJ input data and is used internally by the library for site characterization.

ATYPE_SEDZLJ PROCEDURE DESCRIPTIONS: The "atype_sedzlj" class exposes the following four procedures:

1. *do_initialize*: This routine performs the input of SEDZLJ specific data needed to initialize and drive the SEDLZJ sediment bed mechanics. Two files are read by this routine:

    - BED.SDF: Contains bed properties. Example file in Appendix A.

    - ERATE.SDF: Contains SEDflume data. Example file in Appendix B.

Refer to SEDLZJ documentation for variable definitions.

2. *do_bed_mechanics*: This routine performs the SEDZLJ sediment bed mechanics. This routine is called within the host model's time-marching loop.

3. *do_update_capfs*: This routine is called before the "do_bed_mechanics" routine to update SEDZLJ with current host model information (water depths, shear stress, etc.)

4. *do_update_host*: This routine is called to update the host model with SEDZLJ type information after a call to "do_bed_mechanics" has been made.

CMS-SEDLIBJ ADAPTATION: A derived (subclass) type named "type_sedzlj_cms" was created to drive SEDLIBJ from CMS. Type "type_sedzlj_cms" inherits directly from type "atype_sedzlj". This creates a type hierarchy that facilitates linking SEDLZJ into CMS. The purpose of this type is to encapsulate data and procedures that CMS needs access to execute the SEDZLJ sediment bed processes sub-model. This type is also responsible for driving existing CMS sediment transport routines with SEDLIBJ sediment fluxes ("type_sedzlj_cms" procedures

explicitly call CMS transport routines). This type is conceptualized as belonging with the CMS source code and is separate from the SEDLIBJ library.

```
type, extends(atype_sedzlj), public :: type_sedzlj_cms

logical :: is_active = .false.

logical :: use_cms_transport = .true.

real(kind=zpp), public :: bed_porosity

type(type_constituent_sl), allocatable :: sl_constituents(:)

type(type_constituent_bl), allocatable :: bl_constituents(:)

integer, allocatable :: active_layers(:)

contains

procedure, public :: do_bed_dzdt

procedure, private :: do_bound_c

procedure, public :: do_extern_reset

procedure, public :: do_extern_update

procedure, public :: do_initialization

procedure, public :: do_read_sed_card

procedure, public :: do_transport

procedure, public :: do_sed_distribution

procedure, public :: do_sed_init

procedure, public :: do_update_capfs

procedure, public :: do_update_host

!<CMS TYPE TRANSPORT [Combined bedload & suspended load]>

procedure, private :: do_cms_coeffsourcesink_c

procedure, private :: do_cms_transport

procedure, private :: do_cms_transport_constituents

!</CMS TYPE TRANSPORT>

!<EFDC TYPE TRANSPORT [Separate bedload and suspended load]>

procedure, private :: do_efdc_boundary_bl
```

```
procedure, private :: do_efdc_boundary_sl

procedure, private :: do_efdc_coeffsourcesink_bl

procedure, private :: do_efdc_coeffsourcesink_sl

procedure, public :: do_efdc_compute_bedload_params

procedure, private :: do_efdc_transport

procedure, private :: do_efdc_transport_constituent_bl

procedure, private :: do_efdc_transport_constituent_sl

!</EFDC TYPE TRANSPORT>

end type type_sedzlj_cms
```

Figure 1 displays the class inheritance and collaboration diagrams for the SEDLIBJ interface types described herein.

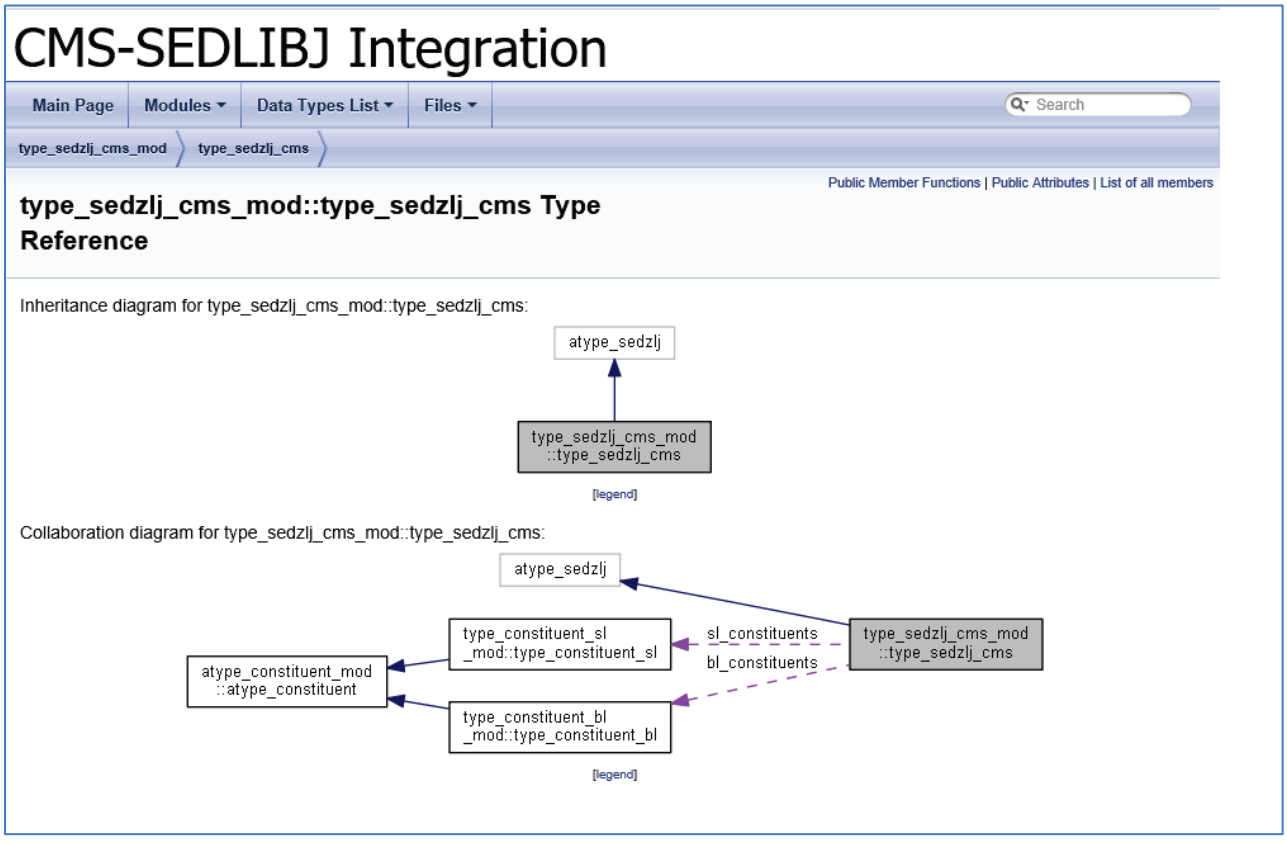

Figure 1. Doxygen generated "atype_sedzlj", "type_sedzlj_cms" class relationship diagram.

TYPE_SEDZLJ_CMS DATA DESCRIPTIONS: The "type_sedzlj_cms' class defines six data objects:

1. *is_active*: Boolean object indicating that SEDZLJ is active.

2. *use_cms_transport*: Boolean object. True specifies suspended load/ bedload transport is being computed simultaneously. False specifies suspended load and bedload transport are computed separately.

3. *bed_porosity*: Holds bed porosity for sediment modeling domain.

4. *sl_constituents*: Array of "type_constituent_sl" objects containing suspended load concentration data when suspended load and bedload transport are computed separately.

5. *bl_constituents*: Array of "type_constituent_bl" objects containing bedload concentration data when suspended load and bedload transport are computed separately.

6. *active_layers*: An integer array containing current active layer index.

TYPE_SEDZLJ_CMS PROCEDURE DESCRIPTIONS: The "type_sedzlj_cms" class defines 14 procedures necessary for driving SEDLZJ from CMS:

1. *do_bed_dzdt*: Updates bed elevation.

2. *do_bound_c*: Checks boundary conditions.

3. *do_extern_reset*: Resets "external data" when doing iterative transport.

4. *do_extern_update*: Updates "external data" after doing iterative transport.

5. *do_initialization*: Initializes "type_sedzlj_cms" object. Wraps base class procedure "do_initialize".

6. *do_read_sed_card*: Reads SEDZLJ cards in CMS control file and posts to "extern data".

7. *do_transport*: Performs transport of sediment classes with SEDZLJ flux estimates.

8. *do_sed_distribution*: Determines sediment class distribution for CMS.

9. *do_sed_init*: Initializes the external data store.

10. *do_update_capfs*: Updates "type_sedzlj_cms" object with host info. Overrides base class procedure.

11. *do_update_host***:** Updates host with "type_sedzlj_cms" object info. Overrides base class procedure

12. *do_cms_coeffsourcesink_c*: Computes coefficients used in LHS of sediment transport equation.

13. *do_cms_transport*: Perform transport with CMS "combined" suspended-load and bedload.

14. *do_cms_transport_constituents*: Performs sediment constituent transport by calling CMS transport routines.

ANCILLARY INTERFACE TYPE DEFINITIONS: Additional interface-enabling derived types for implementing separated (as opposed to combined) suspended and bedload sediment transport are described below:

1. type_constituent: A "base" class data object defined in the SEDLIBJ library. This serves as a transportable sediment constituent data store. See Appendix C for source code listing.

2. type_constituent_bl: A "subclass" data object that inherits from the type "type_constituent" (the base class). This serves as a transportable bedload sediment constituent. See Appendix D for source code listing.

3. type_constituent_sl: A "subclass" data object that inherits from the type "type_constituent" (the base class). This serves as a transportable suspended load sediment constituent. See Appendix E for source code listing.

CMS INPUT OPTION ADDITIONS: Two options were added to the CMS control deck:

1. SEDZLJ_SEDIMENT_TRANSPORT: Specify "ON" to enable SEDZLJ sediment sub-model. Note that user should enable SEDLZJ on the native sediment sub-models at one time – they are mutually exclusive.

2. SEDZLJ_BEDLOAD_SEPARATE: Specify "ON" to enable separate transport of suspended load and bedload. Specify "OFF" to enable combined transport of suspended load and bedload. Default is "OFF". CMS native sediment transport combines both suspended load and bed load transport within one transport equation. The option allows for suspended load and bedload transport to be independently computed (only when running with SEDZLJ enabled). Note that enabling this option moves the now separated suspended and bedload concentrations to the" type_constituent_sl" and "type_constituent_bl" concentration objects in place of the CMS native "Ctk" data array.

CMS TRANSPORT EQUATION MODIFICATIONS: The native CMS sediment transport equation (Sanchez et al. 2014) is written as

$$\frac{\partial}{\partial t}\left(\frac{hC_{tk}}{\beta_{tk}}\right)+\frac{\partial\left(hU_jC_{tk}\right)}{\partial x_j}=\frac{\partial}{\partial x_j}\left(\gamma_s h\frac{\partial(r_{sk}C_{tk})}{\partial x_j}\right)+\alpha_t\omega_{sk}(C_{tk*}-C_{tk}), \quad (1)$$

where

h = water depth $[m]$
U = depth-averaged velocity $[m/s]$
q_{tk} = total-load mass transport
$C_{tk}(=q_{tk}/(Uh))$, actual depth-averaged total-load sediment concentration [kg/m3] for size class k

C_{tk*} = equilibrium depth-averaged total-load sediment concentration [kg/m3] for size class k
β_{tk} = total-load correction factor [-]
r_{sk} = fraction of suspended load in total load for size class k [-]
γ_{sk} = horizontal sediment mixing coefficient [-]
α_t = total-load adaptation coefficient [-]
ω_{sk} = sediment fall velocity [m/s]

Linking CMS with SEDLZJ requires a modification to the CMS native transport Equation (1) as follows:

$$\frac{\partial}{\partial t}\left(\frac{hC_{tk}}{\beta_{tk}}\right)+\frac{\partial(hU_jC_{tk})}{\partial x_j}=\frac{\partial}{\partial x_j}\left(\gamma_s h\frac{\partial(r_{sk}C_{tk})}{\partial x_j}\right)+SS_k \quad (2)$$

where:

SS_k = SEDZLJ net sediment flux

CMS BED-CHANGE EQUATION MODIFICATIONS: The native CMS bed-change equation is written as:

$$\rho_s(1-\rho'_m)\left(\frac{\partial z_b}{\partial t}\right)=\alpha_t\omega_{sk}(C_{tk}-C_{tk*})+\frac{\partial}{\partial x_j}\left(D_s q_{bk}\frac{\partial z_b}{\partial x_j}\right) \quad (3)$$

where:

ρ_s = Sediment density [kg/m3]
ρ'_m = Bed porosity [-]
D_s = Empirical bed-slope coefficient [-]
z_b = Bed elevation with respect to the vertical datum [m]
$q_{bk}\left(=hUC_{tk}(1-r_{sk})\right)$ Bed-load mass transport rate magnitude [kg/m/s]

SEDZLJ linkage to CMS modifies equation (3) to

$$\rho_s(1-\rho'_m)\left(\frac{\partial z_b}{\partial t}\right)=SS_k+\frac{\partial}{\partial x_j}\left(D_s q_{bk}\frac{\partial z_b}{\partial x_j}\right). \quad (4)$$

REFERENCES

Hamrick, J. M. 1992. "A Three-Dimensional Environmental Fluid Dynamics Computer Code." *Theoretical and Computational Aspects*. Williamsburg, VA: The College of William and Mary.

Sanchez, A., W. Wu, H. Li, M. Brown, C. Reed, J. Rosatti, and Z. Demirbilek, 2014. *Coastal Modeling System – Mathematical Formulation and Numerical Methods*. ERDC/CHL TR-14-2. Vicksburg, MS: US Army Engineer Research and Development Center.

Thanh, P., M. Grace, and S. James. 2008. *Sandia National Laboratories Environmental Fluid Dynamics Code: Sediment Transport Manual*, SAND2008-5621. Livermore, CA: Sandia National Laboratories.

Appendix A: Sample SEDZLJ BED.SDF File

```
# VAR_BED ISSEDT Bedload IHTSTRT LAYMAX ISEDTIME NCOHSED IMORPH IFWAVE ICONSOL #
 2 1 1 0 7 1 2 1 0 0

# ITBM NM_SED NSCMAX Array Parameters #
 8 5 9

# ZBSKIN (>0 sets Zo in [um]) DEP_COEFF TAUCONST [dynes/cm^2] ISSLOPE #
 -2.0 1.00 0.0 0

# D50[K] _____________________________________________________D50 of Size Class [um]
 2.5 15.0 100.0 700.0 2750.0

# SGR[K] ____________________________________Specific gravity of sediment size classes
 2.65 2.65 2.65 2.65 2.65

# WWS[1:NCOHSED] _____________Settling speeds of cohesive sediment size classes(mm/s)
 0.0  0.1372

# TCRSUS[K] _________________________________Critical Shear for Suspension [dynes/cm^2]
 0.01  1.6 1.60 10.5 75.8

# TCRE[K] ___________________________________Critical Shear for Erosion [dynes/cm^2]
 0.01  1.6 1.54 3.40 19.4

# SCLOC[NSICM] ______________________________________Sediment Bed Size (um) Tables
 2.0 222.0 432.0 1020.0 2400.0 2600.0 3360.0 6000.0 8520.0

# TAUCRITE[NSICM] _____________Critical Shear for Erosion of Bed Surface [dynes/cm^2]
 0.50 2.27 2.96 4.17 5.88 6.07 6.72 8.48 9.75

# ENRATE[NSICM,ITBM] ____________Erosion rates for active and deposited Layers [cm/s]
 1.00E-09 5.97E-05 5.97E-04 5.96E-03 5.95E-02 5.94E-01 5.94E+00 5.93E+01
 1.00E-09 5.97E-05 5.97E-04 5.96E-03 5.95E-02 5.94E-01 5.94E+00 5.93E+01
 1.00E-09 3.65E-04 2.16E-03 1.27E-02 7.49E-02 4.42E-01 2.61E+00 1.54E+01
 1.00E-09 2.01E-04 1.14E-03 6.51E-03 3.71E-02 2.11E-01 1.20E+00 6.85E+00
 1.00E-09 1.12E-05 8.40E-05 6.28E-04 4.70E-03 3.51E-02 2.63E-01 1.96E+00
 1.00E-09 7.94E-06 6.01E-05 4.55E-04 3.45E-03 2.61E-02 1.98E-01 1.50E+00
 1.00E-09 2.11E-06 1.67E-05 1.31E-04 1.04E-03 8.17E-03 6.44E-02 5.08E-01
 1.00E-09 2.03E-08 1.67E-07 1.44E-06 1.24E-05 1.07E-04 9.28E-04 8.02E-03
 1.00E-09 1.19E-09 2.75E-09 1.69E-08 1.47E-07 1.33E-06 1.22E-05 1.11E-04
```

Appendix B: Sample SEDZLJ ERATE.SDF File

```
# Incore Default_core_id #
 2 1
# [Core 1] Layer Thicknesses For all Cores [per layer] #
 5.0 5.0 5.0 5.0 5.0 5.0 50.0
# Nonerodible Sediment - Critical Shear Stress (dynes/cm^2) [per layer] #
 999.0 999.0 999.0 999.0 999.0 999.0 999.0
 # Bulk Density (g/cm^3) [per layer] #
 1.92 1.92 1.92 1.92 1.92 1.92 1.92
# Particle Size Distribution [row[layer],[column[per modeling size class]] #
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
# ERATE_A [per layer] #
 0.00155 0.00048 0.00052 0.00062 0.00032 0.0000378 0.0000378
# ERATE_N [per layer] #
 2.17 2.88 2.72 2.56 2.85 2.78 2.78
# Initial Bed Erosion Rates [row[itbm], column[per layer]] #
 0.0
 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09
 2.0
 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09
 4.0
 7.330E-04 7.330E-04 7.330E-04 7.330E-04 7.330E-04 7.330E-04 7.330E-04
 8.0
 4.230E-03 4.230E-03 4.230E-03 4.230E-03 4.230E-03 4.230E-03 4.230E-03
 16.0
 2.450E-02 2.450E-02 2.450E-02 2.450E-02 2.450E-02 2.450E-02 2.450E-02
 32.0
 1.430E-01 1.430E-01 1.430E-01 1.430E-01 1.430E-01 1.430E-01 1.430E-01
 64.0
 8.390E-01 8.390E-01 8.390E-01 8.390E-01 8.390E-01 8.390E-01 8.390E-01
 128.0
 4.960E+00 4.960E+00 4.960E+00 4.960E+00 4.960E+00 4.960E+00 4.960E+00
# [Core 2] Layer Thicknesses For all Cores [per layer] #
 5.0 5.0 5.0 5.0 5.0 5.0 50.0
# Riverine Sediments - Critical Shear Stress (dynes/cm^2) [per layer] #
 2.82 2.83 2.82 5.80 5.47 4.88 6.62 14.20 999.0
# Bulk Density (g/cm^3) [per layer] #
 1.92 1.92 1.92 1.92 1.92 1.92 1.92
# Particle Size Distribution [row[layer],[column[per modeling size class]] #
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
 0. 0. 0. 10. 90.
# ERATE_A [per layer] #
 0.00155 0.00155 0.00155 0.00048 0.00052 0.00062 0.00032 0.0000378 0.0000378
# ERATE_N [per layer] #
 2.17 2.17 2.17 2.88 2.72 2.56 2.85 2.78 2.78
# Initial Bed Erosion Rates [row[itbm], column[per layer]] #
 0.0
 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09
```

```
2.0
1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09 1.000E-09
4.0
3.139E-02 2.601E-02 2.257E-02 2.156E-02 1.000E-09 1.000E-09 1.000E-09
8.0
1.413E-01 1.915E-01 1.487E-01 1.271E-01 1.199E-01 1.000E-09 1.000E-09
16.0
6.357E-01 1.410E+00 9.800E-01 7.498E-01 8.648E-01 8.413E-02 8.413E-02
32.0
2.861E+00 1.038E+01 6.457E+00 4.422E+00 6.235E+00 5.778E-01 5.778E-01
64.0
1.287E+01 7.639E+01 4.254E+01 2.607E+01 4.495E+01 3.969E+00 3.969E+00
128.0
5.794E+01 5.623E+02 2.803E+02 1.538E+02 3.241E+02 2.726E+01 2.726E+01
```

Appendix C: SEDLIBJ TYPE "type_constituent" DEFINITON

```
type, public :: atype_constituent

integer :: index = -1

real(kind=zpp), allocatable :: conc(:) ! constituent conc at time level "n"

real(kind=zpp), allocatable :: conc1(:) ! constituent conc at time level "n-1"

real(kind=zpp), allocatable :: conc2(:) ! constituent conc at time level "n-2"

real(kind=zpp), allocatable :: dx(:) ! Conc Gradients in "x" dir

real(kind=zpp), allocatable :: dy(:) ! Conc Gradients in "y" dir

end type atype_constituent
```

Appendix D: SEDLIBJ TYPE "type_constituent_bl" DEFINITON

```
type, extends(atype_constituent), public :: type_constituent_bl

real(ikind), allocatable :: su(:) ! Sub "b" component

real(ikind), allocatable :: su_save(:) ! Per-iteration su

real(ikind), allocatable :: res(:) ! residual

real(ikind) :: tolerance = 1.0e-5 ! Tolerance

real(ikind), allocatable :: dzbl_at_center(:) ! [NCELLS]

real(ikind), allocatable :: dzbl_at_face(:,:) ! [MAXFACES,NCELLS]

contains

procedure size => type_constituent_bl_size

end type type_constituent_bl
```

Appendix E: SEDLIBJ TYPE "type_constituent_sl" DEFINITON

```
type, extends(atype_constituent), public :: type_constituent_sl

real(ikind), allocatable :: su(:) ! Sub "b" component

real(ikind), allocatable :: su_save(:) ! Per-iteration su

real(ikind), allocatable :: res(:) ! residual

real(ikind) :: tolerance = 1.0e-5 ! Tolerance

contains

procedure size => type_constituent_sl_size

end type type_constituent_sl
```