Subdomain ADCIRC+SWAN v.53 User Guide

Technical Report CE-SA-017

Alper Altuntas
Tristan Dyer
Jason Simon
John Baugh

Department of
Civil, Construction, and Environmental Engineering
North Carolina State University
Raleigh, NC 27695

January 21, 2017

Contents

1 Introduction

ADCIRC is a widely used ocean circulation model coupled with SWAN spectral wave model. The computational cost of ADCIRC runs may be prohibitive when many local design and failure scenarios are to be simulated. Subdomain modeling is an exact reanalysis technique for ADCIRC+SWAN that enables the assessment of local *subdomain* changes with less computational effort than would be required by a complete resimulation of the full domain. So long as the subdomain is large enough to fully contain the altered hydrodynamics, multiple local changes may be simulated within it without the need to calculate new boundary conditions (Baugh et al., 2015).

2 Definitions

This section provides definitions of the terms used in this manual. The complete list of definitions of the parameters, input files, and output files used in subdomain modeling is provided in Appendices A.1 and A.2.

Full domain a large geographical area usually requiring substantial computation times. A full domain run, performed once, provides the boundary conditions for a subdomain.

Subdomain a geographical area of interest within the full domain. Multiple subdomain runs with various local changes can be performed using the boundary conditions obtained from a full domain run.

A subdomain modeling run a subdomain run or a full domain run performed to provide the boundary conditions of a subdomain run.

Subdomain modeling control file an input file required for both full domain and subdomain runs. An ADCIRC subdomain modeling run first checks whether a subdomain control file exists in the domain directory. If the file exists, subdomain modeling is activated and the parameters of subdomain modeling are read from the input files. If the file does not exist, ADCIRC performs the simulation as usual.

Shape file a file specifying the location and size of a subdomain grid within the original full domain grid.

Subdomain modeling input files the collection of files used to configure both full domain and subdomain runs. For a subdomain run, input files consist of a shape file, a control file and a boundary conditions file. For a full domain run, the only input file is a subdomain modeling control file.

3 Python Scripts

Python scripts (`gensub.py`, `genfull.py`, `genbcs.py`, `genbcs4swan.py`, `remap.py`) allow users to extract the information from full domain input files and output files, and generate subdomain input files including a nodal attributes file, grid and boundary information file, subdomain control file, and boundary conditions file. Users provide only the parameters that configure the subdomain modeling approach and the locations of the subdomain and full domain directories.

3.1 `gensub.py`

`gensub.py` is used to create subdomain input files. This script reads the subdomain shape file and full domain input files, and creates the following input files in the subdomain directory:

- subdomain control file: `[subdomain dir]/fort.015`
- subdomain nodal attributes file: `[subdomain dir]/fort.13`
- subdomain grid information file: `[subdomain dir]/fort.14`
- subdomain mapping files: `[subdomain dir]/py.14*`

The script also copies the following files from the full domain directory to the subdomain directory if it is a coupled ADCIRC+SWAN simulation:

- subdomain SWAN control file: `[subdomain dir]/fort.26`
- subdomain SWAN initial input file: `[subdomain dir]/swaninit`

3.2 `genfull.py`

`genfull.py` is used to create the full domain control file. This script reads in the subdomain grid information file and mapping files, and creates the control file in full domain directory:

- full domain control file: `[fulldomain dir]/fort.015`

If it is a coupled ADCIRC+SWAN run, the script creates a text file that lists the coordinates at which the two-dimensional spectral outputs are to be recorded as boundary conditions for subdomains, and modifies the full domain `fort.26` file (SWAN control file) accordingly to instruct SWAN to record the spectra at the locations listed in the text file.

3.3 `genbcs.py`

`genbcs.py` is used to generate ADCIRC boundary conditions file for a subdomain grid. This script reads in full domain output files, and creates the boundary conditions file in the subdomain directory:

- subdomain boundary conditions file: `[subdomain dir]/fort.019`

3.4 `genbcs4swan.py`

`genbcs4swan.py` is used to generate SWAN boundary conditions files for a subdomain grid. Note that each boundary node has its own SWAN boundary condition file. This script reads in full domain SWAN output files, and creates the boundary conditions files in METIS partition directories of a subdomain. The script also modifies the partitioned `fort.26` files to include the SWAN boundary forcing commands. This script must, therefore, be executed after a subdomain is preprocessed using `adcprep`.

- SWAN boundary conditions file: `[subdomain dir]/PE*/bc*.019`

3.5 `remap.py`

If a local change, e.g., refinement, applied to a subdomain alters node numbering, `remap.py` may be used to update the nodal mapping file (`py.140`) of the subdomain. This script must be executed before subdomain boundary conditions files are generated using `genbcs.py` and `genbcs4swan.py`.

4 Overview of the Approach

The construction of a subdomain model in ADCIRC consists of four main steps. First, subdomain grid input files and control files are generated. Second, a full domain ADCIRC run is performed. Third, subdomain boundary conditions file is generated using full domain output files. Finally, subdomain ADCIRC run is performed. Figure 1 summarizes the main steps of the subdomain modeling approach, and Table 1 lists the required input files and generated output files for each of the Python scripts. The following section describes each of these steps in detail, including the usage of Python scripts.

5 Detailed Instructions

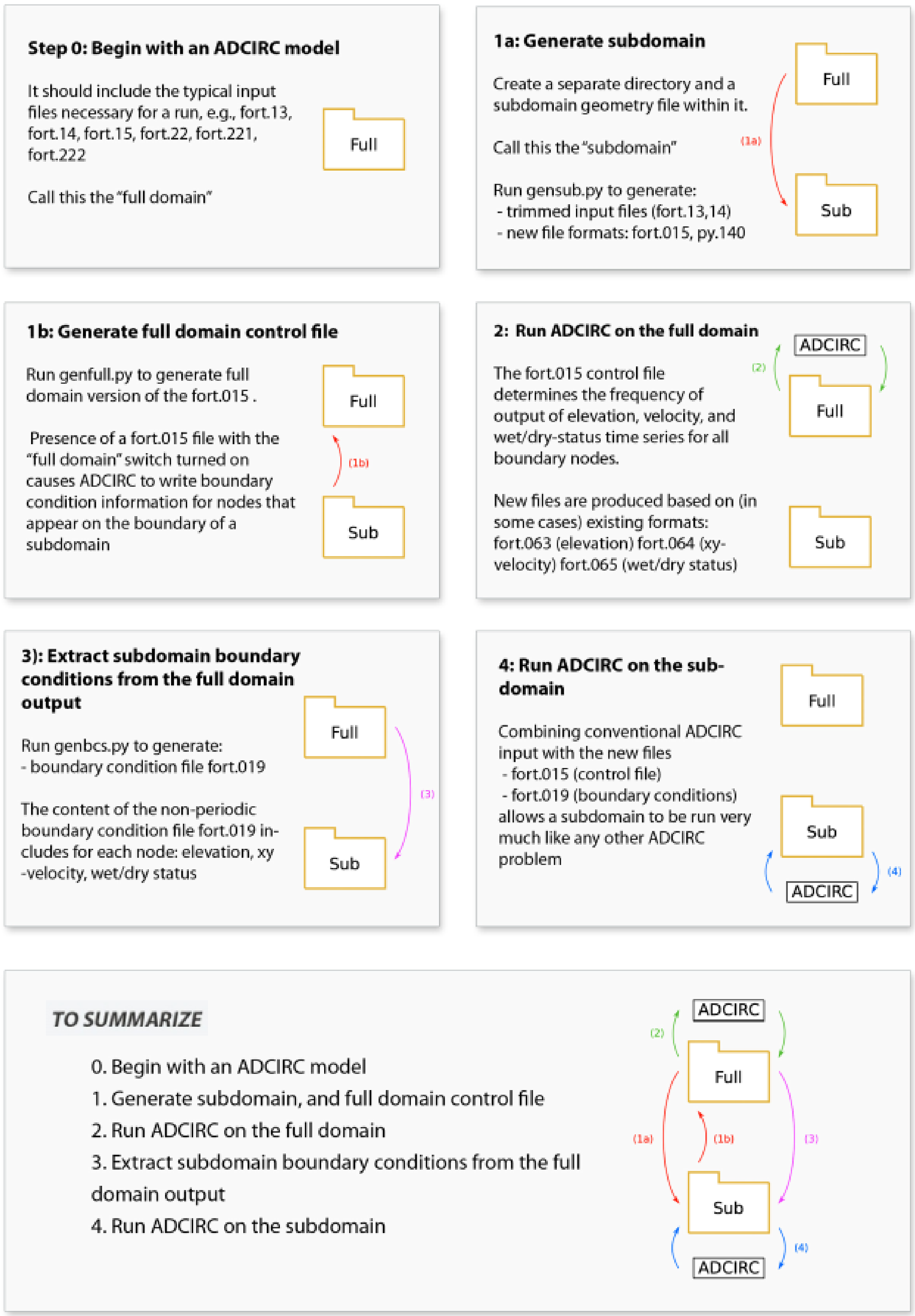
5.1 Generate Subdomain

First, create an empty subdomain directory. The Python script `gensub.py` requires a shape file to extract the subdomain grid from a full domain grid. Create the shape file using a text editor. A shape file contains the coordinates and the size of a subdomain. Two types of subdomain can be extracted from a full domain: circular and elliptical. For a circular subdomain, name the shape file "`shape.c14`", and for an elliptical subdomain, name the shape file "`shape.e14`".

The format of a shape file for a circular subdomain (`shape.c14`) consists of two lines: the coordinates of the center of the subdomain and the radius.

`shape.c14`

```
x  y
r
```



Step 0: Begin with an ADCIRC model

It should include the typical input files necessary for a run, e.g., fort.13, fort.14, fort.15, fort.22, fort.221, fort.222

Call this the "full domain"

1a: Generate subdomain

Create a separate directory and a subdomain geometry file within it.

Call this the "subdomain"

Run gensub.py to generate:
- trimmed input files (fort.13,14)
- new file formats: fort.015, py.140

1b: Generate full domain control file

Run genfull.py to generate full domain version of the fort.015 .

Presence of a fort.015 file with the "full domain" switch turned on causes ADCIRC to write boundary condition information for nodes that appear on the boundary of a subdomain

2: Run ADCIRC on the full domain

The fort.015 control file determines the frequency of output of elevation, velocity, and wet/dry-status time series for all boundary nodes.

New files are produced based on (in some cases) existing formats: fort.063 (elevation) fort.064 (xy-velocity) fort.065 (wet/dry status)

3): Extract subdomain boundary conditions from the full domain output

Run genbcs.py to generate:
- boundary condition file fort.019

The content of the non-periodic boundary condition file fort.019 includes for each node: elevation, xy -velocity, wet/dry status

4: Run ADCIRC on the sub-domain

Combining conventional ADCIRC input with the new files
- fort.015 (control file)
- fort.019 (boundary conditions)

allows a subdomain to be run very much like any other ADCIRC problem

TO SUMMARIZE

0. Begin with an ADCIRC model
1. Generate subdomain, and full domain control file
2. Run ADCIRC on the full domain
3. Extract subdomain boundary conditions from the full domain output
4. Run ADCIRC on the subdomain

Figure 1: Summary of the work flow for subdomain modeling

Table 1: Work Flow of Subdomain Modeling Approach

	Grid	Description

1a	**subdomain**	**Generate Subdomain**
		Python script: `gensub.py`
		requires: `[fulldomain]/fort.{13,14}`
		`[subdomain]/shape.*14`
		generates: `[subdomain]/fort.{015,13,14}`
		`[subdomain]/py.14*`

1b	**fulldomain**	**Generate Full Domain Control File** *Python script:* `genfull.py` *requires:* `[subdomain]/{fort.14, py.14*}` *generates:* `[fulldomain]/{fort.015}`
2	**fulldomain**	**Run ADCIRC on the full domain** *Python script:* - *requires:* `[fulldomain]/fort.{015,13,14...}` *generates:* `[fulldomain]/fort.065`
3a	**subdomain**	**Extract Subdomain Boundary Conditions for ADCIRC** *Python script:* `genbcs.py` *requires:* `[fulldomain]/fort.065` *generates:* `[subdomain]/fort.019`
[**3b**]	**subdomain**	**Extract Subdomain Boundary Conditions for SWAN** *Python script:* `genbcs4swan.py` *requires:* `[fulldomain]/PE*/spec2d.63` *generates:* `[subdomain]/PE*/bc*.019`
4	**subdomain**	**Run ADCIRC on the subdomain** *Python script:* - *requires:* `[fulldomain]/fort.{015,019,13,14...}`

The format of a shape file for an elliptical subdomain (`shape.e14`) consists of three lines: the coordinates of the first focal point, the coordinates of the second focal point, and the width of the ellipse.

`shape.e14`

```
x1 y1
x2 y2
w
```

Once the shape file is saved in the subdomain directory, use the Python script `gensub.py` to create subdomain input files: `fort.015`, `fort.14`. If the full domain has a nodal attributes file (`fort.13`); the script also generates a `fort.13` file for the subdomain. Additionally, if it is a coupled ADCIRC+SWAN run, the script copies the SWAN input files, `fort.26` and `swaninit`, to the subdomain directory. The usage of `gensub.py` is as follows:

- To generate the subdomain input files:

```
$ python gensub.py fulldomainDir subdomainDir
```

Generate `fort.15` *file*

Note: With the merging of subdomain modeling branch into the ADCIRC trunk, the namelist `subdomainModeling` is introduced to ADCIRC. Previously, subdomain modeling was activated if `fort.015` existed in the working directory. In addition to this, ADCIRC now requires `subdomainModeling` namelist to be included in both full domain and subdomain `fort.15` files. This is done by adding the following line to the end of `fort.15` file:

```
&subdomainModeling subdomainOn=T /
```

The subdomain model parameter and periodic boundary condition file (`fort.15`) should be generated by the user. Copy the full domain `fort.15` file to the subdomain directory. Modify the copied `fort.15` file to ensure that the file is compatible with the subdomain run. Required changes to a subdomain `fort.15` include:

- Set `NBFR` (total number of forcing frequencies) to 0.
- Remove lines that define periodic forcing frequencies. Since the boundaries of the subdomain are forced using a boundary conditions file, periodic forcing is not necessary.
- Remove coordinates of any recording stations that are outside of the subdomain grid.
- Update the number of recording stations.

Generate Meteorological Files

If the full domain has any meteorological forcing files (e.g., `fort.22`, `fort.221`, `fort.222`, and so on) that define wind velocity and atmospheric pressure on rectangular (lat/lon) grid(s), e.g., Best Track files and OWI files, symbolically link these files in the subdomain directory. Meteorological files that define wind velocity and atmospheric pressure at each grid node, e.g., NWS=1, are required to be scaled by the user.

Generate full domain control file

The only subdomain modeling input file for an ADCIRC-only full domain run is the control file, `fort.015`.[1] The Python script `genfull.py` is used to create the control file of a full domain run. For ADCIRC+SWAN runs, the script also creates a text file, `swanStations.txt`, that lists the coordinates at which the two-dimensional spectral outputs are to be recorded as boundary conditions for subdomains, and modifies the full domain `fort.26` file (SWAN control file) accordingly to instruct SWAN to record the spectra at the locations listed in the text file. This script prompts user to enter the directories of predefined subdomains and the subdomain modeling parameter `NSPOOLGS`, i.e., the sampling rate. Note that, unlike ADCIRC boundary conditions, SWAN boundary conditions are recorded and enforced at every "SWAN timestep".

The subdomain modeling approach in ADCIRC introduces a new output file based on existing formats: `fort.065`. The sampling rate and reported nodes in this file is determined by the user. The Python script `genfull.py` allows users to specify the boundary nodes of the subdomains as the output nodes. Having an additional set of output files allows ADCIRC to report the data of the boundary nodes of a subdomain at a greater frequency while using less disk space. There are two options for creating a `fort.015` file using `genfull.py`:

1. Assign the boundary nodes of previously created subdomains as output nodes. This option requires much less disk space, but it only provides the output data for subdomain grids generated prior to the full domain run. Boundary node numbers of subdomains are automatically mapped to full domain node numbers and recorded in the `fort.015` file in the full domain directory. Any number of subdomains may be specified.

2. Assign all the full domain nodes as output nodes. This option may lead to excessively large output files, especially when the recording frequency is high. However, the availability of data for each node means that subdomains can later be generated anywhere in the full domain, providing greater flexibility.

The script first asks user if predefined subdomain(s) will be provided. By entering "`n`", users can configure full domain run to record boundary conditions for every node in the full domain grid. If user types in "`y`", the script then asks for the subdomain directories in succession. Once the user enters all the directories, the list of subdomains can be finalized by entering "`done`". The script then asks user to type in the parameter `NSPOOLGS`, the number of timesteps at which information is written to the new output file `fort.065`.

Usage:

```
$ python genfull.py fulldomainDir
```

Note: Add the following line at the end of full domain and subdomain `fort.15` files to ensure that subdomain modeling is activated:

```
&subdomainModeling subdomainOn=T /
```

[1]If the control file does not exist in the full domain directory, the ADCIRC run is performed as usual, i.e., additional output files used to extract the boundary conditions of a subdomain will not be generated.

5.2 Run ADCIRC on Full Domain

Perform the full domain ADCIRC (or ADCIRC+SWAN) run to obtain the new set of output files which are used to produce the boundary conditions of the subdomain. Note that the subdomain modeling approach in ADCIRC is activated only if the control file `fort.015` exists in the working directory and the subdomain modeling namespace is included in the `fort.15` file. Both serial and parallel ADCIRC can be used to perform a full domain run. For parallel ADCIRC runs, execute `genfull.py` prior to executing ADCIRC preprocessor `adcprep`.

5.3 Extract Subdomain Boundary Conditions

Remapping boundary node numbers: As long as the topology of boundary nodes remain unaffected, a subdomain grid may be locally refined. If a local change, like refinement, applied to a subdomain alters node numbering, then a remapping must be applied to update the nodal mapping file `py.140`, before generating the boundary conditions files. If local modifications alter the node numbering of a subdomain, remap the node numbers using `remap.py` script.

Usage:

```
$ python remap.py fullDomainDir subDomainDir
```

The script only remaps the subdomain nodes whose coordinates remain unchanged, which is sufficient for the purpose of generating boundary conditions.

Boundary conditions for ADCIRC: The ADCIRC boundary conditions file of a subdomain is generated using `genbcs.py`. This script reads the `fort.065` file of the full domain, and produces a boundary conditions file, `fort.019`, containing time varying elevations, velocities, and wet/dry flags of boundary nodes, inside the subdomain directory. The optional command line parameter `sbtiminc` may be specified to change the sampling rate of this file.[2] If the script encounters `fort.065` files belonging to both parallel and serial full domain runs, it asks user to determine which output to be used for generating the boundary conditions.

Usage:

```
$ python genbcs.py fullDomainDir subDomainDir [sbtiminc]
```

Depending on the size of the grid and the sampling rate, producing the boundary conditions using `genbcs.py` may take some time.

Boundary conditions for SWAN: The SWAN boundary conditions files containing two-dimensional spectral inputs are generated using `genbcs4swan.py`. This script reads the `spec2d.63` files stored in partition directories of the full domain, and produces a boundary conditions file, `bc*.019`, for each boundary node in the METIS partition directory the node belongs to. The script also modifies the `fort.26` file of each METIS partition to include the

[2]`sbtiminc` must be a multiple of `NSPOOLGS`, the number of time steps at which information is written to full domain output files.

SWAN boundary forcing commands for all boundary nodes within the grid partition. This script must, therefore, be executed after the ADCIRC preprocessor `adcprep`, is executed for the subdomain.

Usage:

```
$ python genbcs4swan.py fullDomainDir subDomainDir
```

5.4 Run ADCIRC on Subdomain

The final step of subdomain modeling approach is to run ADCIRC (or ADCIRC+SWAN) on the subdomain. Multiple ADCIRC runs with various local changes to the subdomain model can be performed using the same boundary conditions file, so long as changes in the hydrodynamics do not reach the subdomain boundaries.

6 Example Usage on the Quarter Annular Test Case

In this section, the subdomain modeling approach is applied to the quarter annular test case, a simple example problem available from the ADCIRC Development Group. The quarter annular grid shown in Figure 2 consists of 63 nodes and 96 triangular elements. The outside arc is an open ocean boundary subject to tidal forces, and the remaining sides are closed land boundaries.

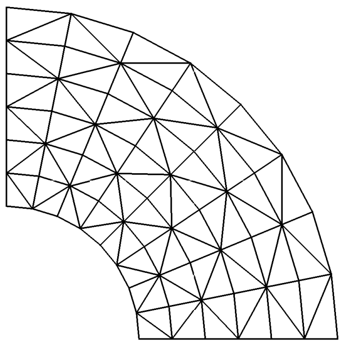


Figure 2: Quarter Annular Test Grid

Using subdomain modeling Python scripts, an elliptical subdomain grid consisting of 16 nodes and 18 elements is extracted from the original quarter annular grid, as shown in Figure 3. Boundary conditions are obtained from the original full run, and are then used for the elliptical subdomain. The steps of the approach are described below.

Generate Subdomain:

1. Create an empty subdomain directory.

2. Create the shape file using a text editor, and save it in the subdomain directory.

```
shape.e14
```

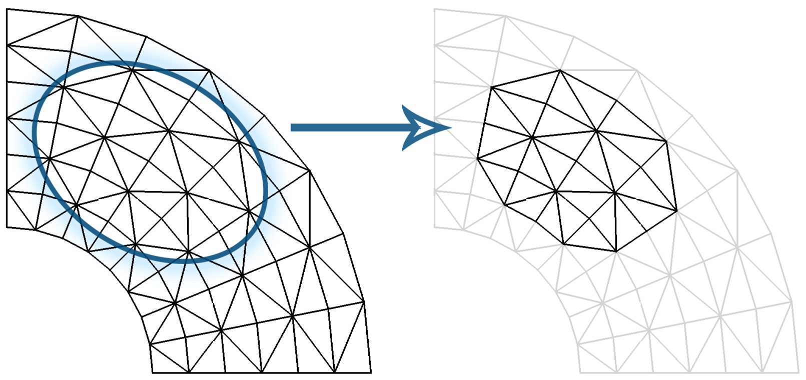


Figure 3: Elliptical Subdomain Grid

```
40824.6 98559.5
98559.5 40824.6
60000
```

3. Navigate to the subdomain directory, and run `gensub.py` to generate the subdomain input files `fort.14` and `fort.015`.[3]

   ```
   $ python [scriptDir]/gensub.py [fulldomain dir] [subdomain dir]
   ```

4. Add the following line to the end of full domain `fort.15` file:

   ```
   &subdomainModeling subdomainOn=T /
   ```

5. Copy the full domain `fort.15` file to the subdomain directory, and modify it as follows[4]:

   - Set `NBFR` to 0.
   - Remove the lines that are related to periodic forcing frequencies.[5]
   - Remove the coordinates of elevation and velocity recording stations.
   - Set the number of elevation and velocity recording stations to 0.

   Note that this example does not require any meteorological file.

6. Move to the full domain directory and run `genfull.py` to generate the full domain control file `fort.015`. Although file sizes are of no concern in this small example, only the boundary nodes of the subdomain are written to the full domain `fort.015` file to illustrate the approach.

[3]Additionally, auxiliary mapping files `py.140` and `py.141` will be created.
[4]The resulting subdomain `fort.15` file is shown in Appendix A.4.
[5]In this example, 12 lines after the line with `NBFR`.

```
$ python [script dir]/genfull.py ./
```

The script asks user if predefined subdomain(s) will be provided. Since a subdomain is created prior to ADCIRC full domain run, type in "`y`", and press `enter`. The script then asks users to type in the subdomain directories. Enter the directory of the QATC subdomain grid. When the script asks for another subdomain directory, enter "`done`" to finalize the list of subdomains. The script then prompts user to type in the parameter `NSPOOLGS`. For this example problem, type "`1`" and press `enter`. Setting `NSPOOLGS` to 1 instructs ADCIRC to record the boundary information for the subdomain run at every timestep.

```
Preparing the full domain at ./ for an ADCIRC run.

Specify predefined subdomains? (y or n). [Default: y]
y
Enter the directory of a subdomain (Type "done" when finished):
../qatc-sub
        Reading fort.14 at ../qatc-sub/
        Reading py.140 at  ../qatc-sub/
Enter the directory of a subdomain (Type "done" when finished):
done

The following subdomains are added:
        ../qatc-sub/

Enter NSPOOLGS (sampling frequency). [Default=100]:
1
        Reading fort.14 at ./
        Writing fort.015 at ./

The full domain is now ready.
```

Run ADCIRC on Full Domain:

Perform the full domain ADCIRC run, either in serial or in parallel.

Extract Subdomain Boundary Conditions:

1. Move to the subdomain directory.
2. Generate the `fort.019` boundary conditions file for the subdomain run using `genbcs.py`.

```
$ python [script dir]/genbcs.py [fulldomain dir] [subdomain dir]
```

Run ADCIRC on Subdomain:

Perform the subdomain ADCIRC run.

Maximum elevation difference

With absolute convergence criteria for the GWCE solver set to 10^{-5}, the absolute error in maximum water surface elevations is on the order of 10^{-7} m. Reducing the convergence criteria further decreases the error.

A Appendix

A.1 Subdomain Modeling Parameters

NOUTGS: Subdomain modeling flag for a full domain run. By setting this parameter to 1 in the full domain control file, the required subdomain boundary conditions for Type-1 are recorded, and by setting this parameter to 2, the required subdomain boundary conditions for Type-2 are recorded. Note: Type-2 runs are for test purposes only.

NSPOOLGS: The number of timesteps at which information is written to the new set of output files; fort.06*.

enforceBN: Subdomain modeling flag for a subdomain run. Type-1 is activated by setting this parameter to 1 in the subdomain control file, and Type-2 is activated by setting this parameter to 2. Note that the required boundary conditions file(s) need to exist in the subdomain directory.

ncbnr: The number of outer boundary nodes of Type-1 subdomain grids to be recorded during a full run.

cbnr: Array of nodes containing the outer boundary nodes of Type-1 subdomain grids to be recorded during a full run.

ncbn: The number of outer boundary nodes of a Type-1 subdomain.

eta2(n): Forced elevation at node n.

etas(n): Forced elevation change at node n.

uu2(n): Forced x velocity at node n.

vv2(n): Forced y velocity at node n.

nodecode(n): Forced wet/dry flag at node n.

A.2 Subdomain Modeling File Formats

A.2.1 `fort.015` - Model control parameters

```
NOUTGS
NSPOOLGS
enforceBN
ncbnr
 for cnode in cbnr:
   cnode
```

A.2.2 `fort.019` - Subdomain boundary conditions file (enforceBN=1)

```
header
nspoolgs, ncbnr, #timesteps
for n in cbnr:
   n
for it in #timesteps:
   it
   for n in cbn:
      n, eta2(n), uu2(n)
      vv2(n), nodecode(n)
```

A.2.3 `fort.065` - Full domain output file (noutgs=1)

```
header
nspoolgs, ncbnr, #timesteps
for it in #timesteps:
   it
   for n in cbnr:
      n, eta2(n), uu2(n)
      vv2(n), nodecode(n)
```

A.3 Modified fort.15 file for the elliptical QATC subdomain

```
QUARTER ANNULAR TEST EXAMPLE 1
ADCIRC V41.03
0                                    ! NFOVER
0                                    ! NABOUT
1                                    ! NSCREEN
0                                    ! IHOT
1                                    ! ICS
0                                    ! IM
1                                    ! NOLIBF
1                                    ! NOLIFA
1                                    ! NOLICA
1                                    ! NOLICAT
0                                    ! NWP
0                                    ! NCOR
0                                    ! NTIP
0                                    ! NWS
1                                    ! NRAMP
9.81                                 ! G
0.005                                ! TAU0
174.656                              ! DT
0.00                                 ! STATIM
0.00                                 ! REFTIM
5                                  ! RNDAY
```

```
2.0                                 ! DRAMP
0.35 0.30 0.35                      ! TIME WEIGHTING FACTORS FOR THE GWCE EQUATION
1.0                                 ! H0
0.0 0.0                             ! SLAM0,SFEA0
0.0025                              ! FFACTOR
0.0                                 ! ESL
0.0                                 ! CORI
0                                   ! NTIF
0                                   ! NBFR
110.0                               ! ANGINN
1 0.0 5.0  3                        ! NOUTE,TOUTSE,TOUTFE,NSPOOLE
0                                   ! TOTAL NUMBER OF ELEVATION RECORDING STATIONS
1 0.0 5.0  3                        ! NOUTV,TOUTSV,TOUTFV,NSPOOLV
0                                   ! TOTAL NUMBER OF VELOCITY RECORDING STATIONS
1 0.0 5.0 1                         ! NOUTGE,TOUTSGE,TOUTFGE,NSPOOLGE
1 0.0 5.0 1                         ! NOUTGV,TOUTSGV,TOUTFGV,NSPOOLGV
1                                   ! NHARFR
M2                                  ! HAFNAM(I)
0.0001405257 1.0 0.0                ! HAFREQ(I=1),HAFF(I=1),HAFACE(I=1)
4.00  5.00  1  0.0                  ! THAS,THAF,NHAINC,FMV
1 1 1 1                             ! NHASE,NHASV,NHAGE,NHAGV
1 512                               ! NHSTAR,NHSINC
1  0  1.E-5  25                     ! ITITER, ISLDIA, CONVCR, ITMAX
```

B Acknowledgments

This material is based upon work supported by the Coastal Hazards Center of Excellence, a US Department of Homeland Security Science and Technology Center of Excellence under Award Number: 2008-ST-061-ND 0001, and a Department of Homeland Security HS-STEM Career Development Grant.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the US Department of Homeland Security.

The authors wish to thank Dr. Rick Luettich and Dr. Brian Blanton for their assistance with ADCIRC and for providing base grids and input files.

C List of Publications and Talks

Altuntas, A. (2012). "Downscaling storm surge models for engineering applications." M.S. Thesis, NC State University, Raleigh, NC, July.
`[http://www.lib.ncsu.edu/resolver/1840.16/8035]`

Baugh, J. W., Altuntas, A., Dyer, T. and Simon, J. (2015) "An exact reanalysis technique for storm surge and tides in a geographic region of interest." Coastal Engineering, 97:60-77.

Baugh, J. W. (2012). "Subdomain modeling." 2012 ADCIRC Workshop, NOAA in Silver Spring, MD, April 23–24.

Baugh, J. W., Rutledge, J., Altuntas, A., and Dyer, T. (2012). "Downscaling storm surge models for engineering applications." Tenth International Conference on Hydroscience & Engineering (ICHE), Orlando, FL, November 4–7.

Baugh, J. W., and Simon, J. S. (2011). "Localized storm surge modeling for engineering analysis and design." ECM 12: Twelfth International Conference on Estuarine and Coastal Modeling, St. Augustine, FL, November 7–9.

Simon, J. S. (2011). "A computational approach for local storm surge modeling." M.S. Thesis, North Carolina State University, Raleigh, NC, June.
`[http://www.lib.ncsu.edu/resolver/1840.16/7179]`

Simon, J. S., and Baugh, J. W. (2011). "Localized modeling of storm surge effects on civil infrastructure using nested meshes." WRRI Annual Conference, Raleigh, NC, March 22–23.