The University of Texas at Austin
Oden Institute for Computational Engineering and Sciences

COMPUTATIONAL HYDRAULICS GROUP
THE UNIVERSITY OF TEXAS AT AUSTIN

pyADCIRC: A Python interface for ADCIRC

Gajanan K. Choudhary, Clint Dawson

Postdoctoral Fellow, Oden Institute for Computational Engineering and Sciences

The University of Texas at Austin

Overview

- Introduction
  - What?
  - Why?
- Implementation
  - How?
  - Sample example
- Application

Introduction

WHAT?

WHY?

Python vs. Fortran

PYTHON

- High-level language
- Faster development cycle
- Slower programs
- One of the fastest growing programming languages [1, 2]

FORTRAN

- Low-level language
- Slower development cycle
- Faster programs
- Legacy codes not going away any time soon

Need for a Python interface

- Python: Vast collection of modern open-source libraries
  - Visualization – Matplotlib, VTK, XDMF
  - Machine learning – PyTorch, TensorFlow
- Fortran: Millions spent in developing, maintaining, and using codes that are now legacy
- <u>Save effort</u> in <u>redeveloping legacy work</u> for use with <u>features of newer languages</u> and vice-versa

pyADCIRC: The ADCIRC Python interface

- A Python package that can be imported into Python
- Allows:
  - Accessing/modifying ADCIRC variables in Python
  - Calling ADCIRC functions from Python
  - Future: Callback functions (calling Python functions from ADCIRC)
- No modifications to existing source code; new files added
  - Modifications recommended for long-term code maintenance

Implementation

HOW?

SAMPLE EXAMPLE

Python-Fortran interface: options

Table 1. Options for building a Python-Fortran interface

Option	Part of CPython	Compiled	Auto-generated	C++ support	Numpy Support
Python/C API*	True	True	False	True	True
ctypes*	True	False	False	False	True
Cython*	False	True	True	True	True
f2py**	False	True	True	True	True

* Requires ISO_C_BINDING module part of Fortran 2003 onward for C, Fortran interoperability

** Intended for Fortran 77/90/95; Other wrappers written on top of f2py also exist

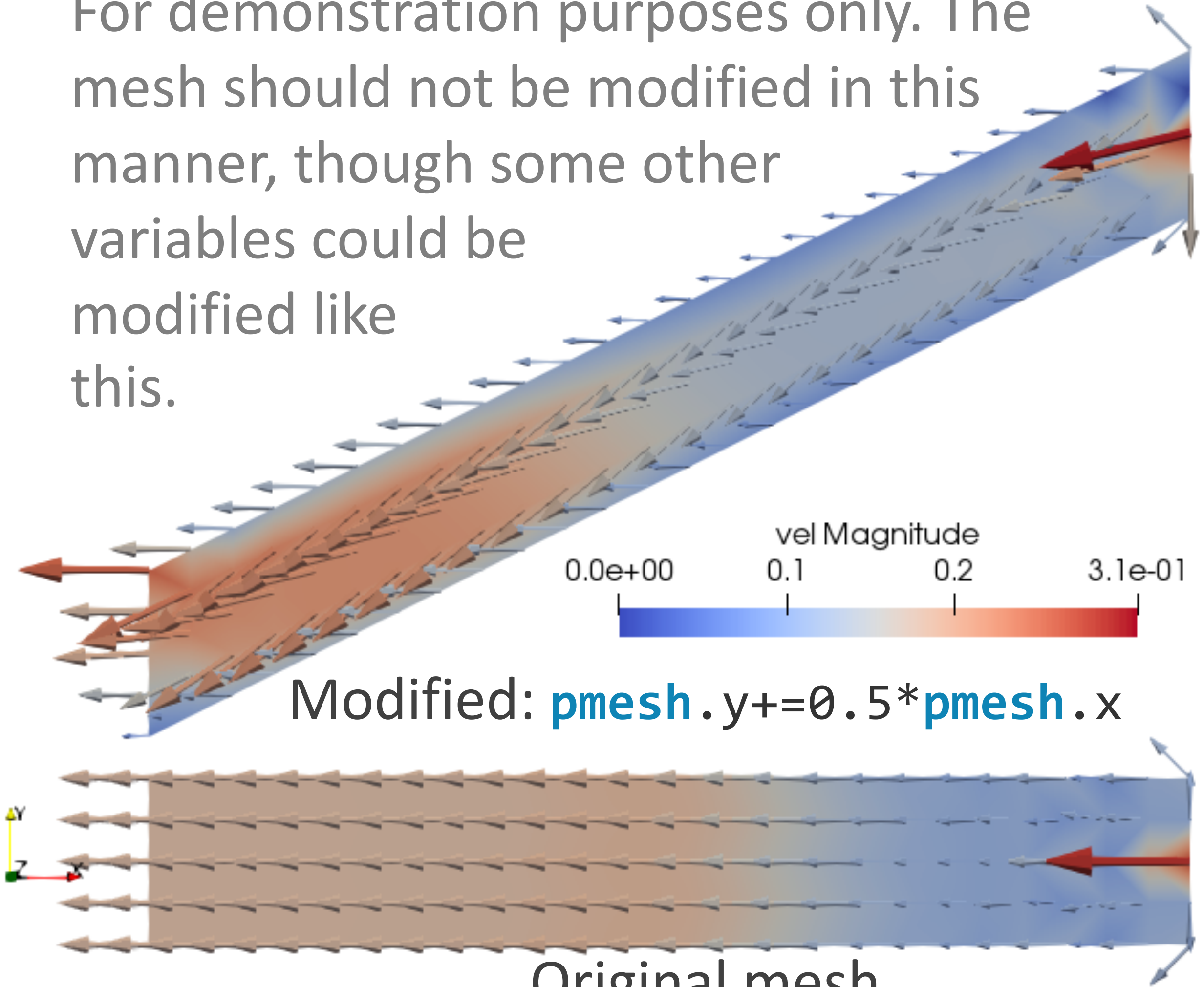
<u>pyADCIRC requires f2py, part of numpy</u>

Example: pyADCIRC

```
In  [1] : from pyadcirc import pyadcirc_mod as pmain # Wrapped adcirc.F
In  [2] : from pyadcirc import pymesh as pmesh       # Wrapped mesh.F
In  [3] : pmain.pyadcirc_init()       # Calls ADCIRC_INIT() - adcirc.F
Out [3] : <Skipping ADCIRC's displayed output for brevity>
In  [4] : print pmesh.y               # Accesses the 'y' variable in mesh.F
Out [4] : array([0. <Skipping for brevity> 8000. 2000. 0.]) # 130 numbers
In  [5] : pmesh.y += 0.5*pmesh.x      # Modify 'y': y=x/2+y
In  [6] : print pmesh.y
Out [6] : array([8000. <Skipping for brevity> 33000. 27000. 25000.])
In  [7] : pmain.pyadcirc_run()        # Calls ADCIRC_RUN() - adcirc.F
In  [8] : pmain.pyadcirc_finalize()   # Calls ADCIRC_FINALIZE() - adcirc.F
```

Result

For demonstration purposes only. The mesh should not be modified in this manner, though some other variables could be modified like this.



Modified: `pmesh.y+=0.5*pmesh.x`

Original mesh

Steps in creating a Python interface

- List modules, variables, and functions needed in Python
- Add f2py directives (comments) to Fortran files
- Compile the source code as a shared library using f2py
- Use it in Python: **import** statement
- Test. Always. Period.

Example: Building a Python interface

```fortran
module adcirc
    integer :: foo
    real*8, allocatable :: bar(:)
    CONTAINS
    subroutine allocateBar(myfoo)
        integer, intent(in) :: myfoo !f2py integer, intent(in):: foo
        foo = myfoo
        allocate(bar(myfoo))
    end
end module adcirc
```

Compilation (Unix):

```
f2py -c \
     -m pyadcirc \
     adcirc.F90
```

Example: Using the Python interface

```
In  [1] : import pyadcirc
In  [2] : print pyadcirc
Out [2] : <module 'pyadcirc' from 'pyadcirc.so'>
In  [3] : print pyadcirc.adcirc.foo
Out [3] : array(0, dtype=int32)
In  [4] : pyadcirc.adcirc.allocatebar(3)
In  [5] : print pyadcirc.adcirc.foo
Out [5] : array(3, dtype=int32)
In  [6] : print pyadcirc.adcirc.bar
Out [6] : array([4.67543053e-310 4.67543072e-310 6.90882188e-310])
```

Application

COUPLING ADCIRC AND GSSHA

COMPARISON WITH ADH-GSSHA COUPLING

Software

- ADCIRC/pyADCIRC:
  - Solves 2D Shallow water equations over oceans and coastal areas
- GSSHA/gsshapython:
  - Solves 2D/1D Diffusive wave equations over inland watersheds
- AdH/adhpython:
  - Solves 2D/3D Shallow water equations over coastal areas

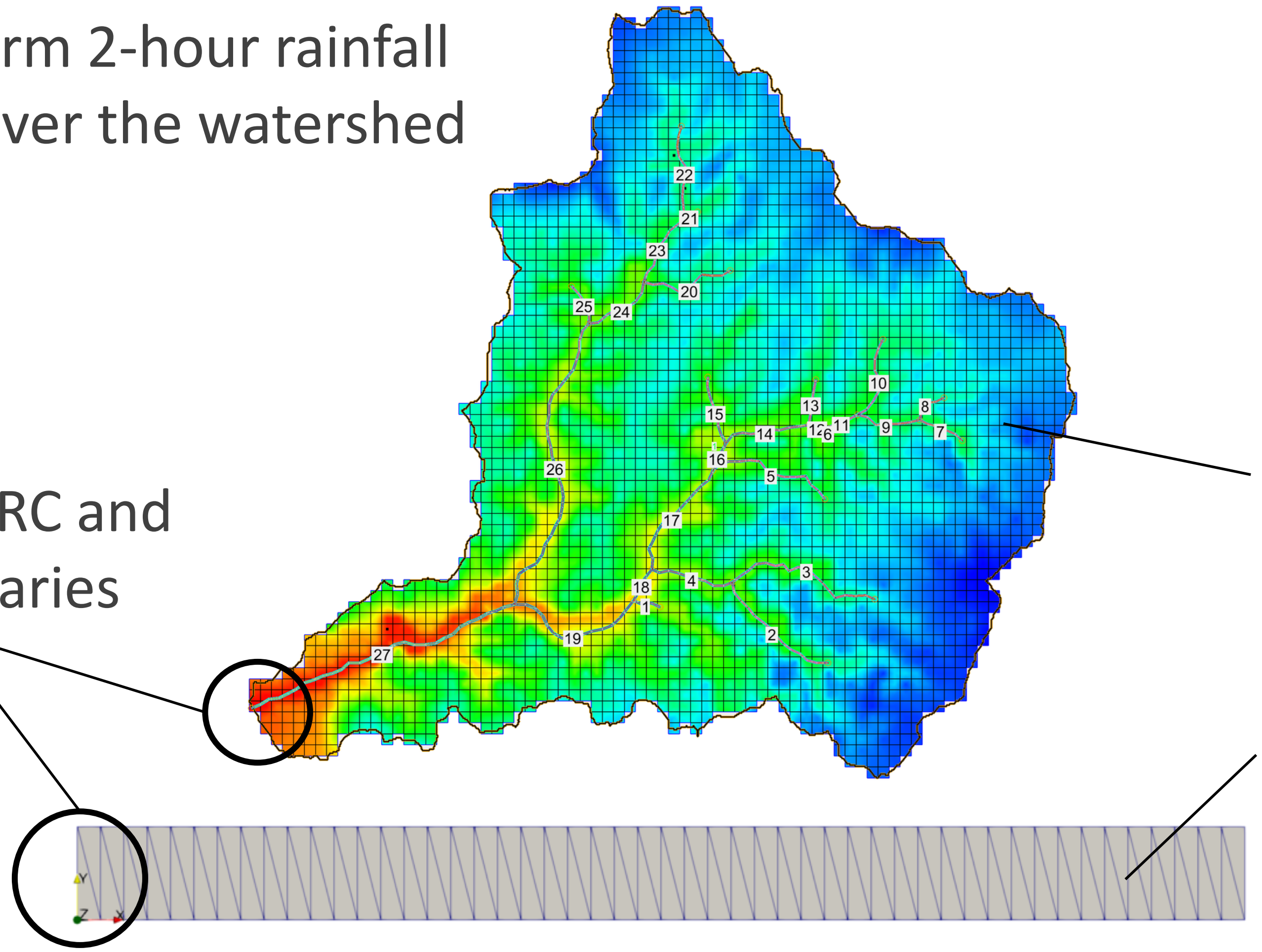
pythoncoupler

- Code for coupling computational software
  - Includes one- and two-way coupling between AdH and GSSHA
  - Includes one- and two-way coupling between ADCIRC and GSSHA
  - Requires AdH, GSSHA, and ADCIRC python interfaces
  - Models may use different time steps, starting/ending times
  - Coupled through <u>in-memory data exchange</u>, no File I/O

Watershed emptying into a 'tank'

Forcing: Uniform 2-hour rainfall of 40mm/hr over the watershed

Coupled ADCIRC and GSSHA boundaries



GSSHA model: Hypothetical watershed

ADCIRC model: Rectangular tank

Fig. 2. Meshes of coupled GSSHA and ADCIRC models

Comparison with other coupled software

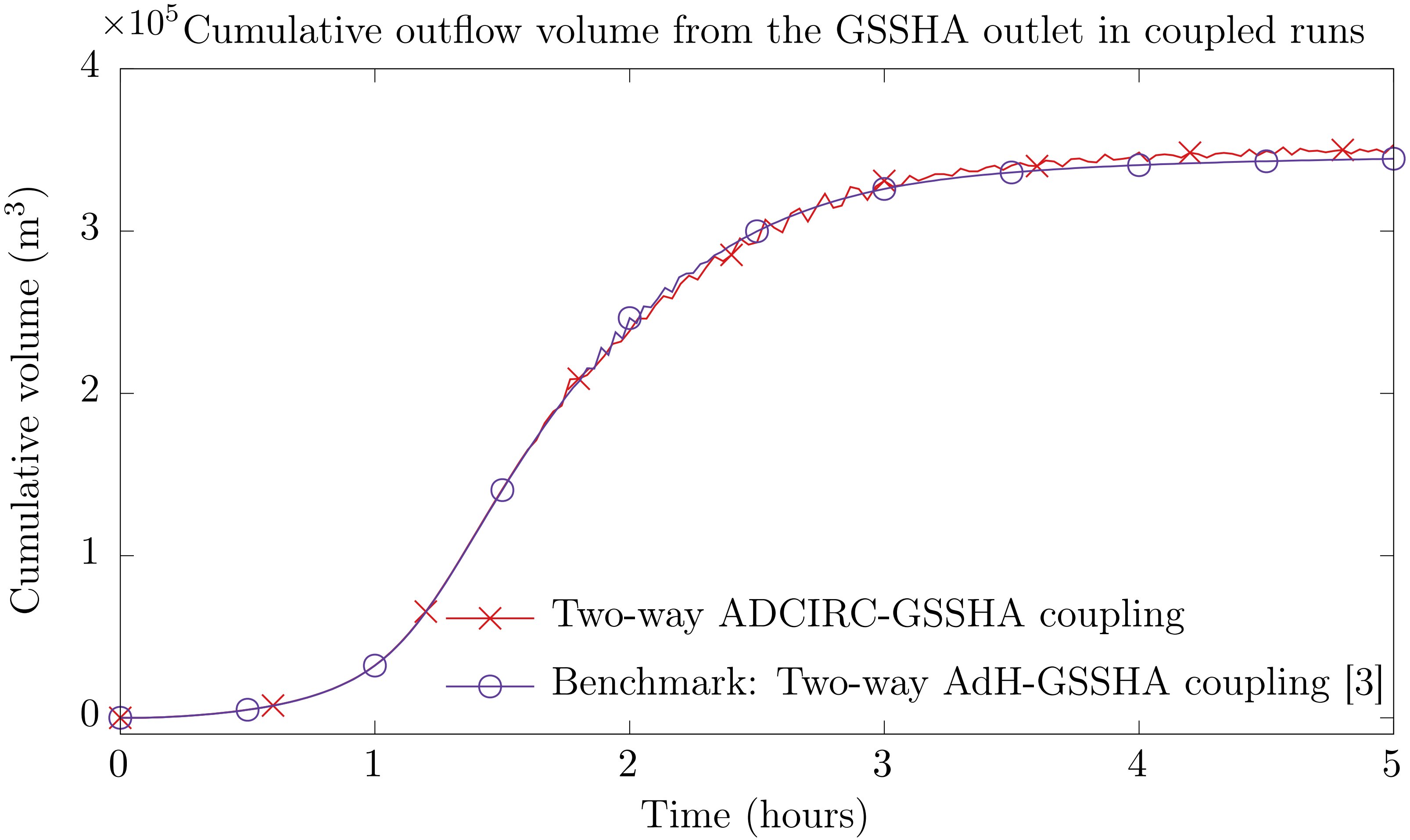


Fig. 3(a). Comparison of ADCIRC-GSSHA coupling against AdH- GSSHA coupling [3]

Comparison with other coupled software

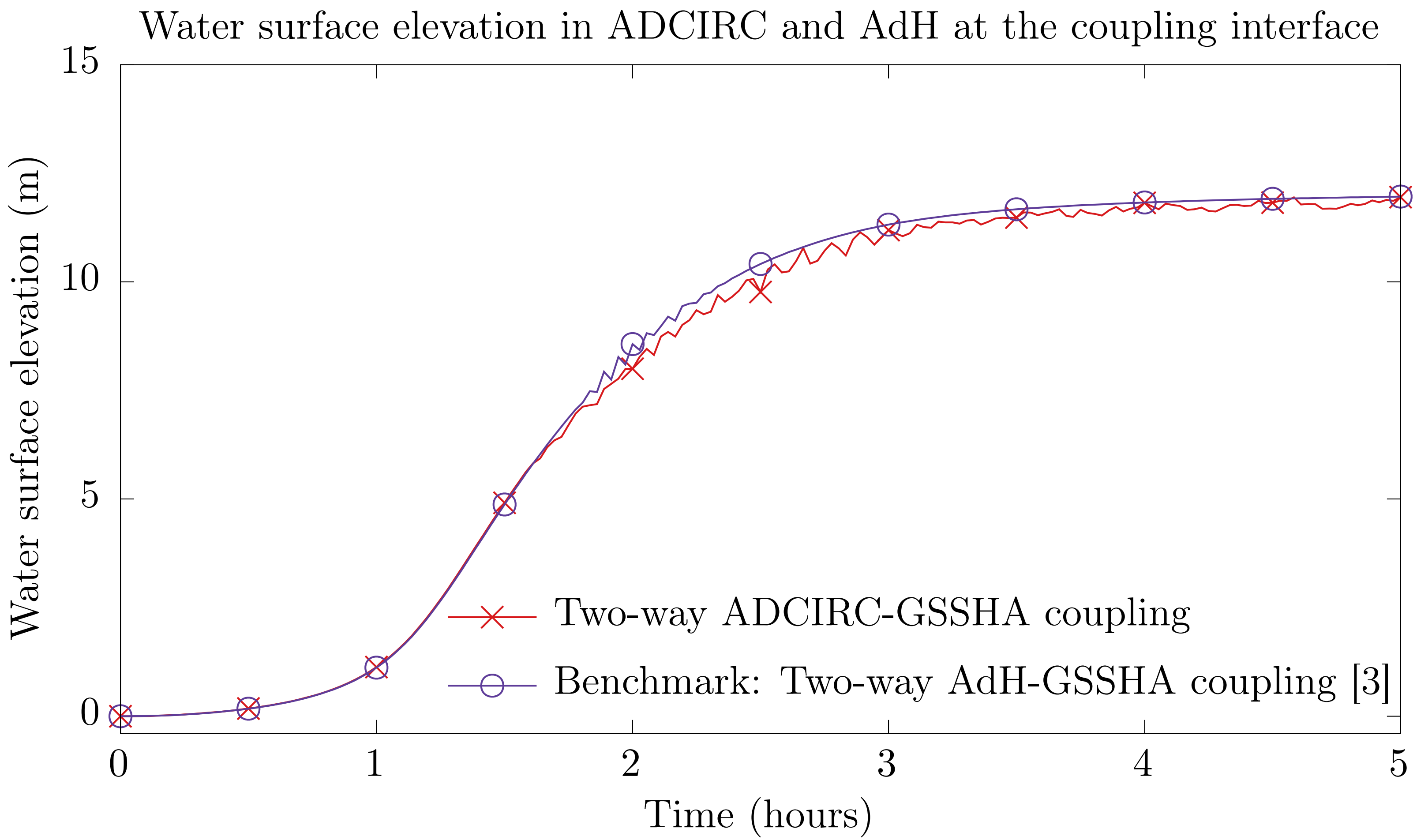


Fig. 3(b). Comparison of ADCIRC-GSSHA coupling against AdH- GSSHA coupling [3]

Compound flooding effect

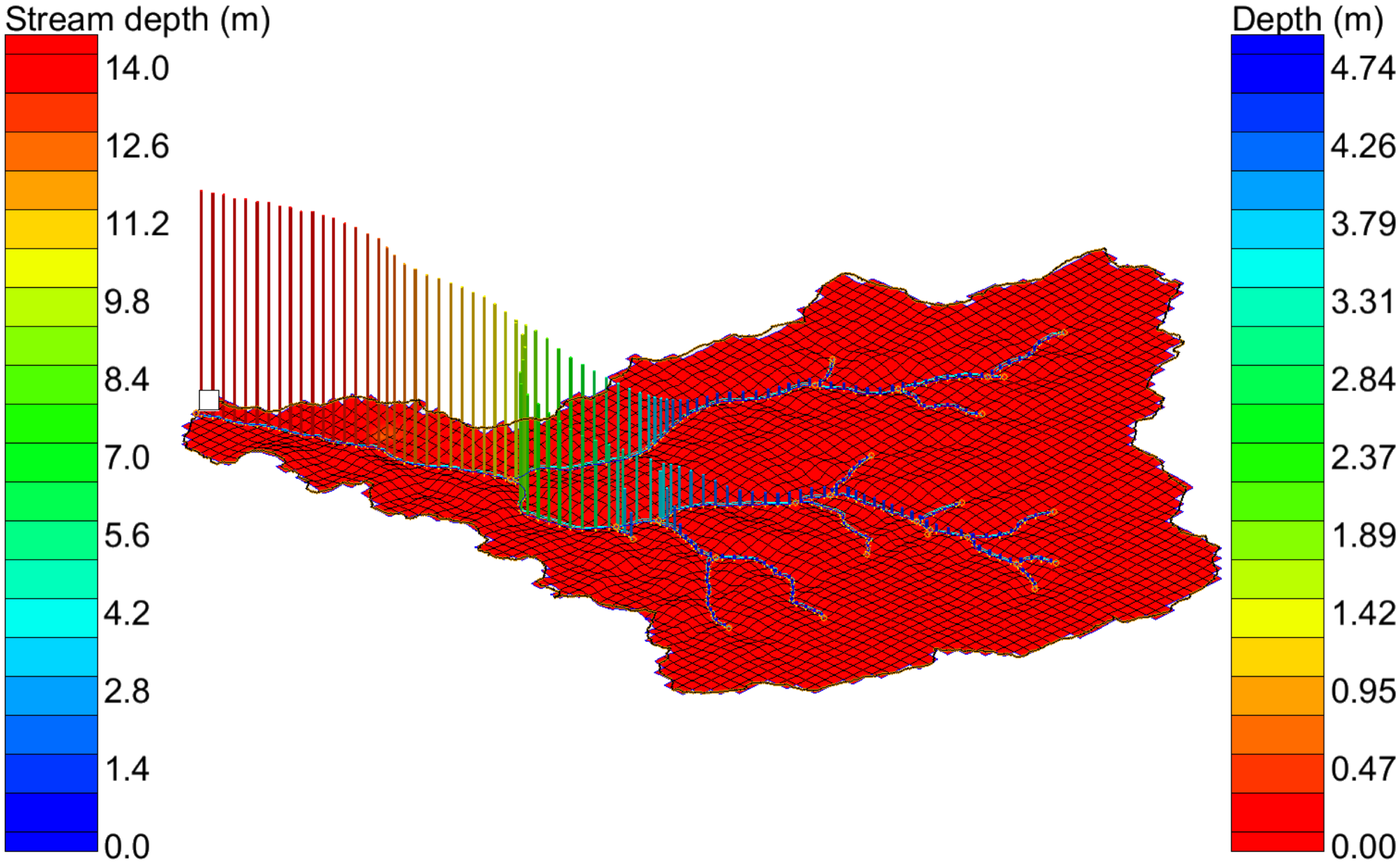


Fig. 4. Flooded watershed at the end of a 5-hour simulation

Conclusion

pyADCIRC: The ADCIRC Python interface

- A Python package that can be imported into Python
- Allows using variables and functions of ADCIRC in Python
- No modifications to existing source code; new files added
- Opens up new avenues of research
  - Multi-software coupling (compound flooding)
  - Machine learning

Future work

- Adding call-back functions
- Unit/integration testing
- Explore other uses of pyADCIRC: Pre/post processing, I/O, machine learning, coupling with other software, etc.
- Github, licensing, and version control considerations

  - Interested? Please e-mail: gajanan@utexas.edu

References

References

[1] TIOBE Programming Community Index. Access link (as of 12/04/2019): https://www.tiobe.com/tiobe-index.

[2] RedMonk Programming Language Rankings for January 2019. Access link (as of 12/04/2019): https://redmonk.com/sogrady/2019/07/18/language-rankings-6-19/.

[3] Choudhary, G.K. (2019). Coupled atmospheric, hydrodynamic, and hydrologic models for simulation of complex phenomena

Thank You!

Appendix

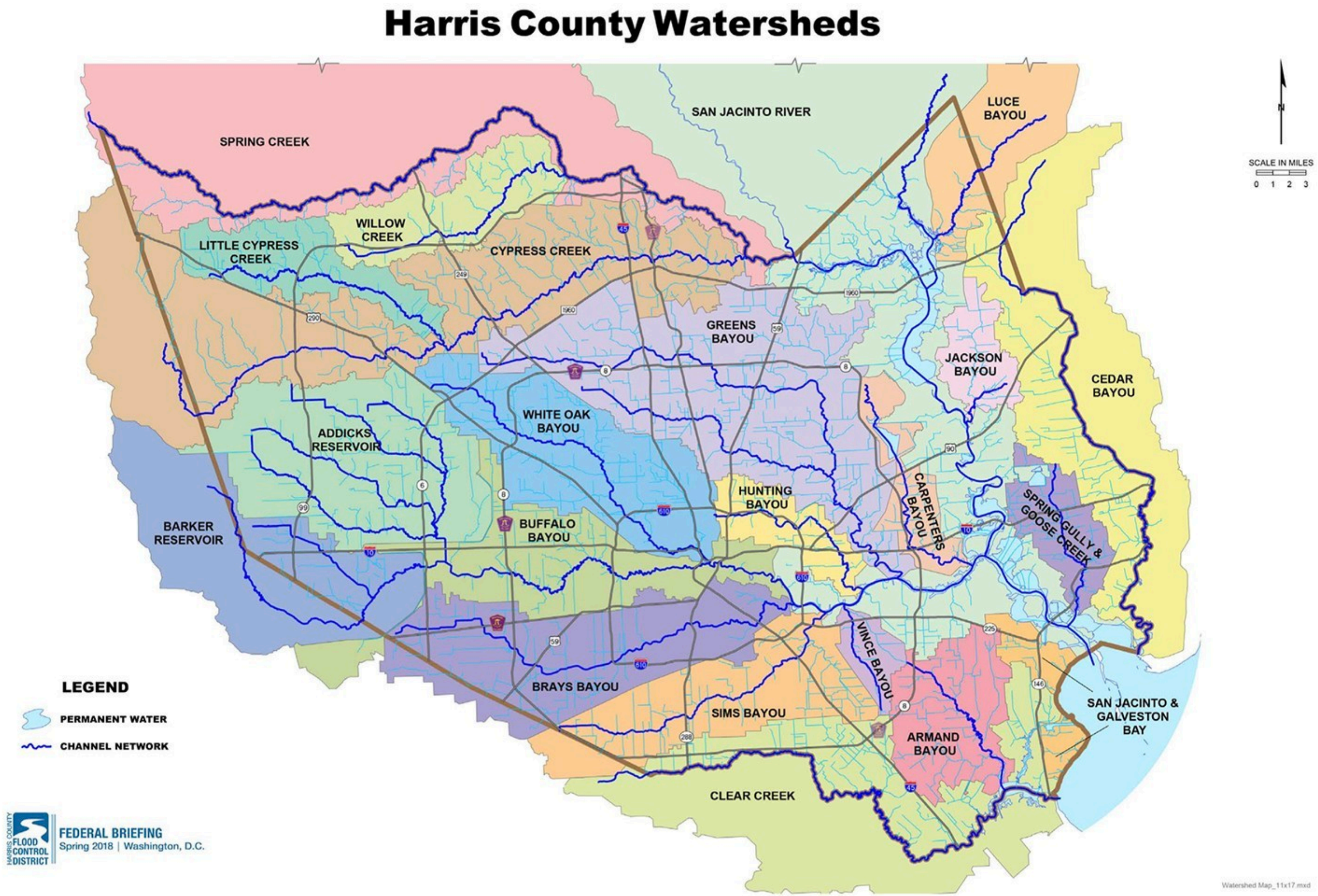
ADH-GSSHA COUPLING RESULTS: HURRICANE HARVEY

REFERENCE [3]

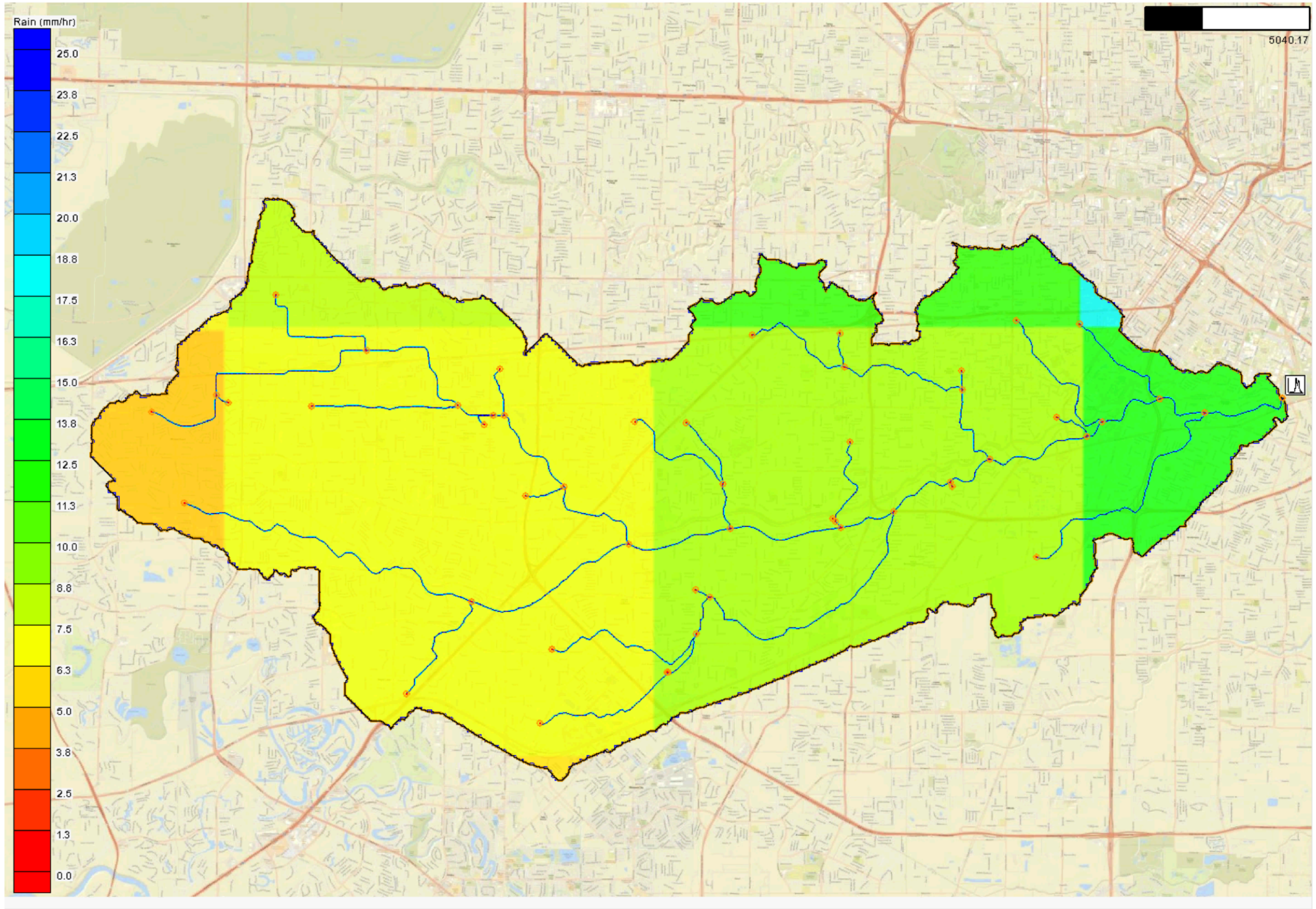
AdH-GSSHA coupling results

- Hurricane Harvey – August 2017
- One of the costliest hurricanes to hit the US coast
- Massive floods
- Attempt: Coupling AdH and GSSHA to simulate Harvey
  - GSSHA: Brays Bayou watershed
  - AdH: Galveston Bay

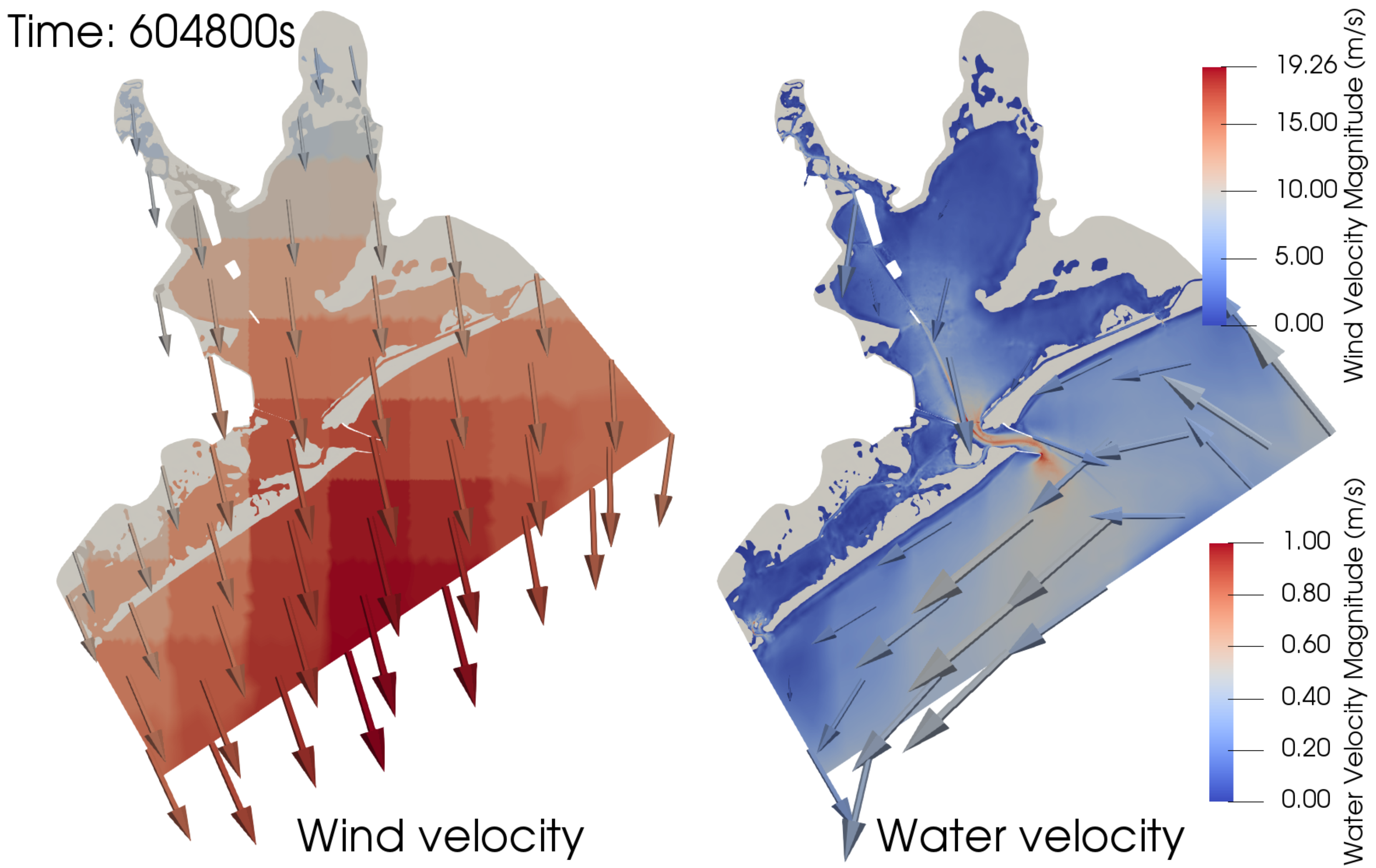
Harris County Watersheds



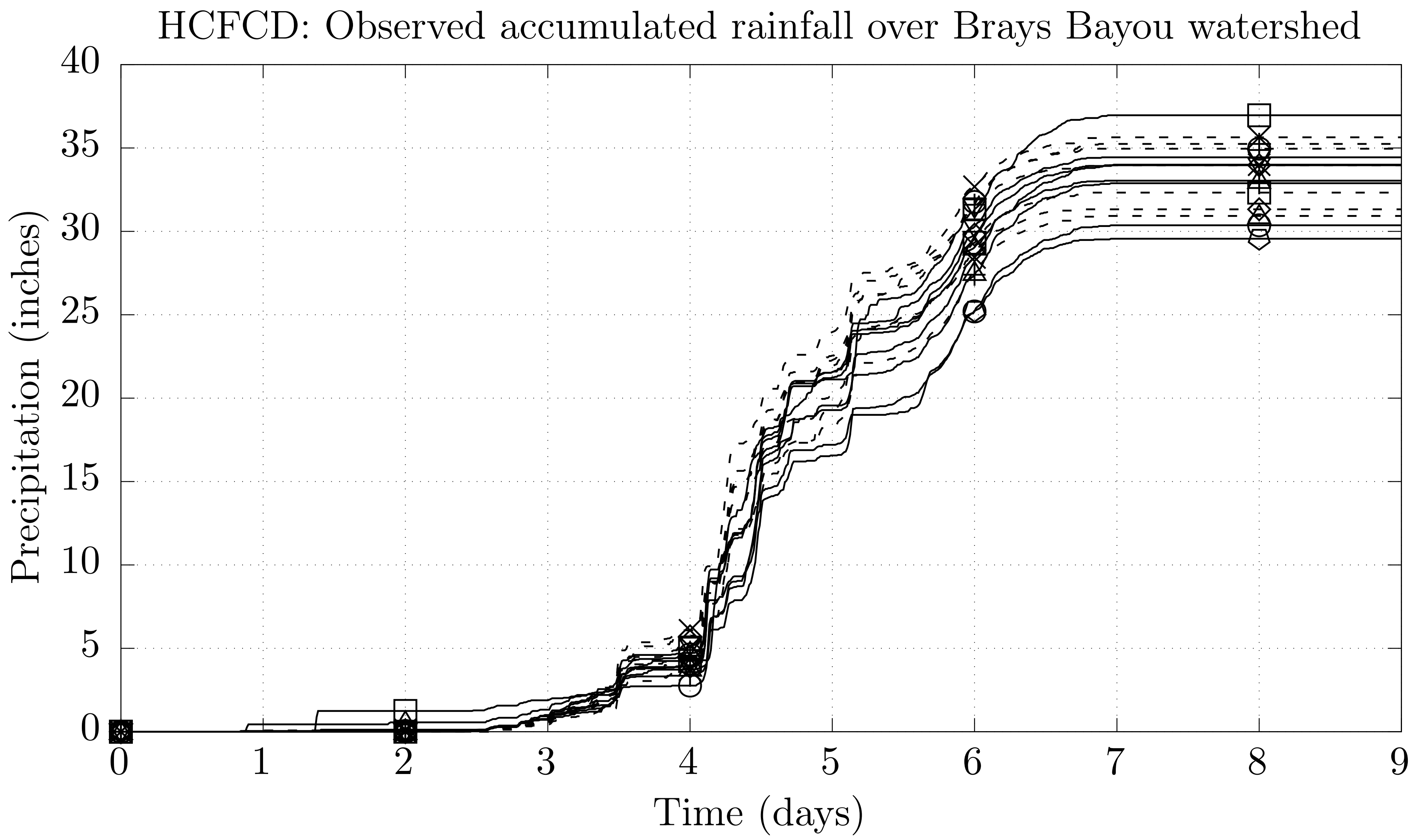
Brays Bayou Watershed model



Galveston Bay model



HCFCD: Observed rainfall during Harvey



AdH-GSSHA coupling