Hydraulic Modeling: Manual Modeling Instructions

January 2018

Matt Nahorniak

The following steps outline the process required to manually run CHaMP hydraulic models, either as batches or for individual models. These instructions cover default models and/or models with non-default flow rates. These instructions do not include steps required to run porous structures within hydro models, which will be covered under separate documentation. The first section of this document goes into some detail about how to run hydro models, while the 2nd section provides a more concise checklist of steps that users may wish to consult one familiar with the process.

Dependencies:

1. AWS instances (EC2 images), or
2. Local Delft
3. **Detailed Instructions**

**Run locally or on Amazon AWS instance**

Before starting, users must choose between running on their local pc or an Amazon AWS instance.

To run locally, a compiled version of Delft3D must be present. The executable file “d\_hydro.exe” should be present in the folder structure “**C:\Matt-SFR Files\Delft3D\Delft3D\_Updated\delft3d\_ohmw\_4.01.00.rc.02\delft3d\win32\flow2d3d\bin**”. If Delf3D is stored using different directory names, the R-Code and corresponding batch files created in the process may not know where to find the executable files.

Note that version Delf3D version **4.01.00.rc.02** is used. This is the version used for all CHaMP modeling development. It’s not clear whether newer releases of the code will function properly using the CHaMP process. An un-compiled version of this code and all accompanying files and documentation, is freely available from Delft3D. CHaMP users will likely find it easier to copy the compiled version of the code directly, maintaining the file structure, rather than re-compile. Even copying the code directly, it’s not always simple to get the code to work the first time. Follow the instructions found in the pdf file “install\_ohmw\_4.01.00.rc.02.pdf” in the directory “**C:\Matt-SFR Files\Delft3D\Delft3D\_Updated**”. Note that the instructions also refer to a user license. This license is for the Delf3D-Flow GUI, which is not used as part of the Delft-3D modeling process, and is an unnecessary step. (Serious users may wish to acquire this license and use the GUII, however. This license is also free from Delft3D. Consult the Delft3D website and documentation).

To run on an Amazon AWS instance, sign in to the South Fork Research (SFR) Amazon AWS account, and launch an AWS instance with the latest Delft and hydro modeling updates. Contact Carol Volk or Matt Reimer for access instructions. We have instances to be launched with all the required Delft3D Flow software, R, the required R-scripts, and related files all set to go

The instance to launch is generally it’s called “Hydraulic Modeling Working Copy\_mm\_dd\_yy”. Use the latest date. For “type” of instance, select “C4.4xlarge”. For all else choose default settings. Connect to remote desktop connection as “Administrator” and use your password stored in the file “windows\_password.txt”. I’m not sure why, but I’ve found it’s sometimes necessary to connect as “administrator” rather than the default name for hydro to run correctly.

Once on the Amazon instance, ensure latest copy of R scripts and related files are downloaded. Source tree is present on the AWS instances. Open Source Tree and check for updates. In general, the latest updates of all required files will be present on the latest AWS image. Updates are frequently made where hard-coded specification of boundary conditions are required. Other than that, the default code is quite stable.

**Required Inputs:**

The following three file are required, and should be in the same directory for each VisitID being modeled.

**DEM.csv**: a .csv file containing bathymetry information. There are three columns: “x” and “y” are the x and y coordinates (x\_albers and y\_albers), and “value” is the z coordinate (the bathymetric elevation).

**WSEDEM.csv**: a .csv file with for the water surface elevation. There are three columns: “x” and “y” are the x and y coordinates (x\_albers and y\_albers), and “value” is the water surface elevation, in meters.

**Thalweg.csv**: a .csv containing points along the thalweg. There are two columns: “x” and “y” are the x and y coordinates (x\_albers and y\_albers). It is not critical whether these points go from downstream to upstream or vice-versa, as the R-code will figure that out based on corresponding water surface elevations at each thalweg point by assuming the water surface elevation decreases, on average, in the downstream direction.

These files can be in any directory of the user’s choosing, but uploading results to the Amazon QA bucket will be easier if the directory structure matches the directory structure of the Amazon QA bucket, after locally creating a folder structure “c:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket” (more info below). For example, the following directory on a local hard drive or AWS instance should be used for VisitID 4766:

**C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket\2017\Asotin\ASW00001-CC-F2P1BR\VISIT\_4766\Hydro\HydroModelInputs**

Pulling these input files from the Amazon AWS bucket can be automated using scripted queries. For example, to download the above files into the directory above, the following batch script can be run:

**aws s3 sync s3://sfr-champdata/QA/2017/Asotin/ASW00001-CC-F2P1BR/VISIT\_4766 "C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket\2017\Asotin\ASW00001-CC-F2P1BR\VISIT\_4766" --exclude "\*" --include "\*\HydroModelInputs\\*"**

This query will automatically generate the needed directory structure on your local PC or the Amazon AWS instance. It’s a good idea to make sure the folder and all sub-folders of “**C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket”** are empty before this step.

Note: There is a separate process for generating and depositing the hydro model input files to the ‘HydroModelInput’ folder on AWS. The input files can be generated from the CHaMP Workbench and copied into the AWS folder, or they can be generated on a local desktop and uploaded to AWS.

While it appears tedious to generate this batch script, this is made easy by use of a pair of excel spreadsheets. One spreadsheet (“CHaMP\_and\_AEM\_Metrics.csv”) contains site names, visitID’s, watersheds, years, etc., as well as VisitID. On another spreadsheed (“Hydro Modeling Data Pull.xlsx”), on the “Data Pull from AWS” worksheet, simply enter the visitID to model in the left most column, and copy/paste the remaining columns and the query will automatically be constructed in column M. This can be done for a single site or a group of sites. Simply copy column M for all rows of interest, open a command line prompt, and paste these values in. If working on an AWS instance, you can do this step on your local PC, then copy the query or queries into a command promp on you AWS instance. (The AWS instances don’t contain MS Excel or any substitute spreadsheet editor, so this needs to be done on your local PC). Note that column F in the spreadsheet does not affect the query. It can be used to add a comment or note about the query.

**Special Note for AEM queries:**

The spreadsheet above generates queries to pull data from the aws bucket sfr-champdata. All CHaMP data are available at this bucket. However, if aem sites are to be modeled, then the value in column H should be changed from “s3://sfr-champdata/QA/” to “s3://sfr-aempdata/QA/”

**CFD\_Site\_List.csv**: The remaining inputs are specified in a file called “CFD\_Site\_List.csv”, which needs to be in the following folder: **“C:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files”.** This file contains the following inputs, with one for each model to be run:



The column “Directory” (not fully shown in the figure above) is the directory containing the three main input files (DEM.csv, WSEDEM.csv, Thalweg.csv). Model is a “Yes” if you want to run a model from this row, Measured.Discharge is the “as surveyed” discharge in cms, “Modeled.Discharge” is the discharge at which you’d like to model the flow. D84 is the measured value of D84. Roughness is the roughness input to Delft3D, currently set at 4 x D84/1000 for all models. Trim.Length describes how many meters is trimmed off the inlet and outlet boundaries for the computational grid. HEV is horizontal eddy viscosity, and is always set at 0.01. (There has been one VisitID ever where we had to increase this to 0.1 to enable a stable numeric solution in Delft3D). DeltaBC is set to zero by default, but can be used to specify a change in water surface elevation at the exit boundary when running at unmeasured flows. For example, if we know the water surface elevation at the exit boundary is .15 meters higher at the modeled discharge relative to the measured discharge, we could input .15 meters in this row. Generally we do not have such information, and we leave this at zero. (We know this is an underestimate at flows higher than default, but we’ve found it’s better to underestimate the exit BC water surface elevation than to over-estimate it).

Again, it may seem tedious to generate this input file, but for the most part it has been automated. There is an R script “**Build\_CFD\_Site\_List.R**” that, for the most part, generates this file automatically. Simply open this file in R (it’s in the directory: C**:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files\R-Code”** and run the entire script. It will search for all input files in the “**C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket**” folders, and pull required data from “**CHaMP\_and\_AEM\_Data.csv**”, and generate “**CFD\_Site\_List.csv”** for the default flow runs for whatever input files that are present. If any information is missing (D84 or discharge, for example) the “Model” column will be “No” for that row. For non-default flow rates, the user can manually enter the discharge to model be editing the “Modeled.Discharge” column. Non-default D84’s are occasionally run, in which case the user must edit both the “D84” and “Roughness” columns (again, setting Roughness to 4 x D84 / 1000).

Note that the Trim.Length column is 2 by default. However, in the directory C**:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files\R-Code** there is a file called “**TrimLengths.csv**” that contains all non-default trim lengths, as determined by trial and error after running nearly all CHaMP sites. These non-default trim lengths will automatically be read and added to the “**CFD\_Site\_List.csv”** file.

**Pre-Processing:**

Once the input files are in the correct locations and “**CFD\_Site\_List.csv”** has been prepared, the R script “Build\_Input\_Files.R” is run. Open this file and ensure the working directory is “C**:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files\R-Code”**, and run the entire code by selecting ctrl-a to select all and ctrl-r to run all.

This code does several things. It generates a batch file (“batchprocess.bat”) that will be used to launch Delft3D-Flow in the next step. It creates another batch file (“vs.bat”) that will eventually be used to convert Delft3D-Flow output to text output that can be read back into R. These two input files are put into the directory “C**:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files”**.

The “build\_input\_files.R” code also creates a series of input files used by Delft3D-Flow. These are stored in a sub-directory of where the input files are located, such as “**C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket\2017\Asotin\ASW00001-CC-F2P1BR\VISIT\_4766\Hydro\HydroModelInputs\S0000\_1867**”. Note the last directory name, which is generated by the R-code. It will begin with either an “S” or and “M”. “S” indicates that the flow rate being modeled is equal to the surveyed flow rate, while an “M” indicates that the modeled flow rate is different from the surveyed flow rate. The remaining 9 characters indicate the flow rate, with a “\_” used instead of a decimal point. These directory names are used when later syncing results with the AWS bucket. The input files in this directory include files for boundary conditions, computation grid point locations, bathymetry, water surface elevation, distribution of upstream discharge, etc. Most have names that begin with “Test” (a relic leftover from when these input files were first being tested for the first time). A meta-data file is also generated to pass information about the run on to the post-processing and later included with the results. There is also an “offset.csv” file that stores an offset between the actual X-Y grid locations and the grid locations used by Delft3D. (For reasons having to do with the number of digits Delft3D uses in X-Y locations, we can’t use the full X-Y locations in the inputs, so we subtract off the a common value from all X-Y locations such that the origin becomes 0,0 in the computational grid. These offsets for X and Y are then added back in post-processing). Finally, there’s an “updated\_thalweg.csv” file that’s included, as the order of points in the thalweg file is sometimes reversed to ensure it always is ordered from upstream to downstream.

Pre-processing can take several minutes or more for each VisitID x Flow combination specified in “**CFD\_Site\_List.csv”**. The pre-processing will proceed automatically for all rows of “**CFD\_Site\_List.csv”** where “Yes” is specified in the “Model” column.

The pre-processing script will also generate a set of four jpg files for each hydro model, which will be stored in a directory of the form “**C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket\2017\Asotin\ASW00001-CC-F2P1BR\VISIT\_4766\Hydro\Results\S0000\_1867”**. Note that a “Results” folder has been created, and a subfolder of the same name structure “**S0000\_1867**” has been created. If more than one flow is modeled for the same VisitID, multiple sub-folders under “Results” will be generated. The jpg images should be examined before proceeding, especially the “**Boundary\_Conditions.jpg**” image, to ensure adequate boundary conditions have been generated. This is the most important step in the QA process. Consult Matt Nahorniak for more information regarding what to look for in this image. Also present are jpg files showing bathymetry, water surface elevation, and water depth. These files are primarily for troubleshooting should errors occur at some point during processing.

It can be tedious, when running large batch jobs, to manually locate all folders storing all QA plots. For this reason, an R-script called “**BC\_jpg\_copy.R**” has been created. This script copies all the QA plots into a single folder for easy viewing. That folder is “**C:\Matt-SFR Files\Hydraulic Modeling\BC Copies**”. Running “BC\_jpg\_copy.R” is optional, and has no effect on the input files or modeling results. When using this script, it’s a good idea to make sure “**C:\Matt-SFR Files\Hydraulic Modeling\BC Copies**” is empty or non-existent before running the script. (This same script can be used again at the conclusion of the modeling process to move additional QA plots generated in post-processing).

**Manual Boundary Condition Over-Rides**

Occasionally, boundary conditions as observed in the file “**Boundary\_Conditions.jpg**” may be inadequate. The two most common issues are having trim lengths that are either too great or too small, or having an inlet or outlet boundary that’s not along the most appropriate edge (north, south, east, or west). See Appendix A for details and examples on what to look for when examining QA plots.

**Trim length override**

However, the user may manually edit the trim length after reviewing the boundary conditions. This change can be made directly in the “CFD\_Site\_List.csv” file. For default jobs, it’s recommended that the change be added to the “TrimLengths.csv” file, and the updated file be checked in to control via SourceTree. Note that the TrimLengths.csv file is also consumed by the aws automated version of the R-code, so these updates, once uploaded to github, will automatically happen with the automated version as well.

**Boundary condition location override**

Manual changes to the boundary condition location (north, south, east, or west) are made as hard code changes in the R-code “Build\_Input\_Files.R”. In the algorithm, both inlet and outlet boundary locations are determined by finding to which grid edge (before trimming) the closest thalweg points is located. To manually over-ride the boundary as determined by the algorithm, we simply set the distance to the desired edge to zero. The distances to the four boundaries are variables called “dnorth”, “dsouth”, “dwest”, and “deast”. For the inlet boundary condition, search the code for the following comment:

**# Inlet boundary manual corrections**

Then add a line under this comment of the following form:

if (site.list$SiteID[k]== "LEM00002-00001B") {dsouth = 0}

This example manually sets the inlet boundary for site LEM00002-00001B to the south boundary. You’ll find plenty of examples in the code under the “# Inlet boundary manual corrections“ comment. You’ll simply have to copy one of them, and change the site name and the cardinal direction. Note that R is case sensitive, so be sure to enter the sitename exactly, and be sure to use lower case letters for called “dnorth”, “dsouth”, “dwest”, or “deast”

The outlet boundary manual corrections can be made exactly as the inlet boundary manual corrections, except that the location in the R-code “Build\_Input\_Files.R” is different – it’s several lines below the inlet manual override. Search for the following comment:

**# Outlet boundary manual corrections**

and then proceed exactly as you would for an inlet boundary condition manual override. For example, adding the following line forces the exit boundary for site CBW05583-086186 to be on the west edge:

if (site.list$SiteID[k]== "CBW05583-086186") {dwest = 0}

If you make any of these hard code changes, please use Sourcetree to upload the edited code to the github repository. Also, contact Matt Reimer (or whoever is in charge of the automated aws process) and have these manual over-rides added to the aws automated version of the code. The automated process uses a separate R-code that runs on Linux, and must be edited separately.

**Running Delft3D-Flow**

Once the pre-processing script has finished, Delft3D-Flow is initiated by running the batch file “batchprocess.bat” located in the directory “**C:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files**”. Simply click on the file. This will launch Delf3D-Flow and proceed to process all jobs in serial. It make take anywhere from a few minutes to several hours or more to process each hydro job. Large batches may take several days to run.

Delft3D-Flow creates several files during execution, which are stored in the same folder as the input files. Consult the Delf3D flow documentation for full details on each of these files. A sometimes useful file (which will be copied to the results folder in post-processing” is “tri-diag.test”, which contains information about the run itself, including warnings and error messages. Generally, the warnings can be ignored. (Many of the warnings pertain to the veracity of the changing flow prior to steady state, but we’re only concerned with the steady state solution. Other warnings may pertain to monitoring stations “outside the computational domain”, which can also be ignored. A warning about modeling occurring “at the equator” can also be ignored – recall we set the origin to 0,0. This only would matter if we were concerned about tidal forces. In general, ignore all warnings and errors unless the solution fails to complete).

If, for some reason, the solution fails to complete, Delft3D-Flow will proceed to the next job automatically.

Delft3D-Flow will, if all goes well, continue un-interrupted. Progress for each individual job can be followed on the command prompt, although users cannot tell from the command prompt which hydro model is currently running. (If you really must know or want to track progress on the complete batch of jobs, you can look in the input files folders – for example: **C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket\2017\Asotin\ASW00001-SF-F4P1\VISIT\_4782\Hydro\HydroModelInputs\M000000\_09** - to see for which jobs Delft3D-Flow has generated output files).

In general, individual batch sizes should not exceed 40 or 50 models, with batch sizes of 25 or less being more common and less cumbersome. It’s not difficult to start multiple aws instances simultaneously, so batch size need not limit total throughput.

Typical hydro models runs take an hour or so per model. Larger sites take slightly longer to run than small sites, and runs at very low flow rates may take significantly longer than runs at higher flow rates. For flow rates less than .0001, the “build\_CFD\_SiteList.R” file will specify “No”, indicating the model will not run. However, this can be over-ridden by manually changing the CFD\_Site\_List.csv file to “Yes” for the “Model” column. Flows of zero, however, will cause an error.

**Post-Processing**

Once Delft3D-Flow has finished all jobs in a batch, post processing can begin. The first step is running the batch file “vs.bat” in the directory “**C:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files**”. This runs the “viewer selector” tool, which converts Delft-3D-Flow output into a series of text files that R can read as inputs. Consult the Delft-3D flow documentation “Viewer Selector User Manual.pdf” for more information. This takes a few seconds for each hydro model generated.

One this step is complete, open and run the script “Post\_Processing.r” in the directory “**C:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files\R-Code**”. This will take several minutes or more for each hydro model generated.

The post processing script generates the full set of hydro model results, and puts them into the results directories created in pre-processing (i.e. of the form “**C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket\2017\Asotin\ASW00001-CC-F2P1BR\VISIT\_4766\Hydro\Results\S0000\_1867**”.

A second QA step should occur at this time. Key files to examine are the jpg files “Velocity\_Magnitude.jpg”, “Depth.jpg”, and “Depth\_Error.jpg”. Consult Matt Nahorniak for details on what to look for in these plots. Again, the script “BC\_jpg\_copy.r” can be used to move all of these files, from all jobs in the batch, to a single folder (“**C:\Matt-SFR Files\Hydraulic Modeling\BC Copies**”) for easy viewing.

**Synching Results to AWS Bucket**

For most jobs, users will want to upload the results to the sfr-champ AWS bucket. This can (and should) be done for all “default” jobs that pass QA. “Default” in this sense includes all jobs at measured flow rates *and* all jobs at modeled flow rates, as long as all other inputs are default. Jobs run with modified DEMs, porous structures, non-default D84 inputs, or any other non-default settings should not be uploaded to the AWS bucket.

First, delete any results for jobs that failed QA. Next, open a command prompt.

For CHaMP results, run the following command:

**aws s3 sync "C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket" s3://sfr-champdata/QA --exclude "\*" --include "\*\Hydro\Results\*"**

If you’re uploading AEM results, run the following command:

**aws s3 sync "C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket" s3://sfr-aemdata/QA --exclude "\*" --include "\*\Hydro\Results\*"**

For convenience, a pair of batch files to run these commands have been created and stored in the folder: “**C:\Matt-SFR Files\Hydraulic Modeling**”. These files are named “Results\_to\_CHaMP\_QA\_bucket.bat” for CHaMP results, and “Results\_to\_AEM\_QA\_bucket.bat” for AEM results. Be sure you’ve deleted any results that failed QA prior to running these scripts! Also, be sure to use the correct script! Do NOT use the CHaMP script for AEM results, or vice-versa!

Results not synced to an AWS bucket should be zipped up and uploaded to Dropbox or otherwise distributed by the user.

At this point, modeling is complete. If running from an AWS server, the instance must be shut down, and users should terminate the instance to stop incurring AWS charges. DO NOT FORGET TO TERMINATE THE AWS INSTANCE! Once the instance is terminated, all files will be lost except results files uploaded to the AWS bucket(s).

1. **Checklist: Step by Step Running the CHaMP Hydraulic Modeling**

If running on an Amazon AWS instance, perform the following steps. If running locally, skip to step 3:

1. Launch AWS instance of AMI with the latest Delft and hydro modeling updates. Generally it’s called “Hydraulic Modeling Working Copy\_mm\_dd\_yy”. Use the latest date.
   1. For “type” of instance, select “C4.4xlarge”. All else choose default settings.

Notes: For this step, access to the SFR AWS account is required. Contact Carol Volk or Matt Reimer for access instructions. We have instances to be launched with all the required Delft3D Flow software, R, the required R-scripts, and related files all set to go.

1. Connect to remote desktop connection
   1. Connect as “Administrator”
   2. Use password from file “windows\_password.txt”

Notes: I’m not sure why, but I’ve found it’s sometimes necessary to connect as “administrator” rather than the default name.

1. Ensure latest copy of R scripts and related files are downloaded
   1. Open Source Tree and check for updates

Notes: In general, the latest updates of all required files will be present on the latest AWS image. Updates are frequently made where hard-coded specification of boundary conditions are required. Other than that, the default code is quite stable.

1. Ensure folders “C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket” is empty or non-existent. As described above, these folders are where the hydro prep folders are to be downloaded from the aws bucket. Later processing steps will automatically generate a list of sites to be modeled based on a search of all hydro prep files present in the subfolders of this folder. Anything left in there inadvertently will also be added to the list of sites, so you’ll want to ensure no unwanted hydro prep files are included in this folder.
2. Download hydro prep data from cmsource, aem, or other appropriate aws bucket
   1. Open in Excel: “CHaMP\_and\_AEM\_Metrics.csv”
   2. Open Excel file: “Hydro Modeling Data Pull.xlsx”
   3. Enter list of VisitID’s to model in first column of “Hydro Modeling Data Pull.xlsx”
   4. Auto fill the remainder of the columns from template row(s)
   5. Copy column of AWS download commands for sites to model from “Hydro Modeling Data Pull.xlsx” (Column M) and paste into a command line prompt. This will pull data from the AWS bucket and put it into the correct directory structure.

Column H in this Excel file specifies which AWS bucket to pull from. Options for column H are: “s3://sfr-aemdata/QA/” for AEM visits, or more commonly “s3://sfr-champdata/QA/” for CHaMP visits.

Note: Do NOT run batches with both AEM visits and CHaMP visits in the same batch. At the data upload step (see below) all data is uploaded to either the AEM or CHaMP QA buckets, and the scripts are not able to mix and match results. You do NOT want to accidentally upload AEM results to the CHaMP QA bucket, or vice versa. This is VERY IMPORTANT!

1. Generate and Check “CFD\_Site\_List.csv”
   1. Run R-code “Build\_CFD\_Site\_List.R”

The file “CFD\_Site\_List.csv” specifies the list of sites to run hydro models for, as well as the measured and modeled discharges, and D84 value, and misc. other values. It also specifies directories where the hydro prep files are located and drives where the results files will be stored. The R script run here automatically builds this file based on the hydro prep files found from step 6, above. It also queries a .csv copy of the CHaMP/AEM data for discharge and D84 values.

This file can be manually edited as needed. If running non-default flow rates, the user may change the “modeled discharge” to a discharge of their choice. If, along with a change in discharge, a known or estimated change in water surface elevation at the downstream exit boundary is available, that can be entered in this file and it will improve the downstream boundary condition significantly. If a manual D84 value is used (over-riding the default value), then the “roughness” value must also be changed manually: set it to D84\*4/1000.

The column “model” simply indicates whether to include a visitID in the hydro model runs. Thus the user may turn off or on individual visit ID’s when running the model.

If editing the file “CFD\_Site\_List.csv”, I f find it easiest to copy and paste “CFD\_Site\_List.csv” file to my local PC, modify columns as needed, then copy and paste “CFD\_Site\_List.csv” back to the AWS PC

1. Run “Build\_Input\_Files.R” on AWS PC (will take hours for more than a few sites)

Notes: This is the key pre-processing step that generates all the input files for Delft 3D flow, as well as batch files and xml files needed. It also generates a suite of QA files.

1. QA check #1: Check boundary conditions jpg file for proper BC

The pre-processing R-script generates a file called “Boundary\_Conditions.jpg” for each visitID being modeled. A simple way to check all of these boundary conditions is to run the R-script “BC\_jpg\_copy.R”. This will copy all of the boundary condition QA files into a single directory to make it easy for users to scroll through them and examine them as a batch. The directory they’ll be stored in is “C:\Matt-SFR Files\Hydraulic Modeling\BC Copies”.

If you are unfamiliar with what to look for when performing QA assessments, contact Matt Nahorniak for training.

If needed and if possible, manually adjust boundary conditions by varying the trim length in “CFD\_Site\_List.csv” and repeat “Run Build Input Files.R” step. Please contact Matt Nahorniak if boundary conditions are changed, or if you’re unable to achieve satisfactory boundary conditions by adjusting the trim length. In cases where manual over-ride of the exit boundary face (north, south, east, or west) is required, this will need to be hard-coded into the pre-processing R script – please contact Matt Nahorniak in this case as well.

1. Run batchfile “batchprocess.bat” in the directory: “C:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files” (Note: may take anywhere from 10 minutes to may hours to run each site; this it can take many days for batches of more for 20-30 sites)
2. Upon completion of the batchprocess.bat script, run the script “vs.bat” in the directory: “C:\Matt-SFR Files\Hydraulic Modeling\R Code to Build Input Files”

This converts Delft 3D output to text output. This step should take only a few seconds per visitID modeled.

1. Run “Post\_Processing.R”

This will may take several hours for batches of 10 or more sites. This script converts the text outputs from Delft 3D Flow into results translated back onto the original DEM grid, and generates the full suite of hydro modeling results files.

1. QA check #2

Users should examine the full suite of QA files. As in QA check #1, users can run the script “BC\_jpg\_copy.R” to copy all QA files into the directory “C:\Matt-SFR Files\Hydraulic Modeling\BC Copies”. Examine each file to ensure the full site is wetted, and there are no outrageous errors obvious for which results should not be used. Again, consult Matt Nahorniak for guidance on the QA process.

If results fail QA, they will need to be removed. Find the VisitID in question from the subdirectories of “C:\Matt-SFR Files\Hydraulic Modeling\champ data from bucket”. Go into the “results” folder for the model in question, and delete the results. Also contact Matt Nahorniak with the visitID and details about why the model in question failed.

1. Upload results to appropriate bucket

Be careful to upload results to the appropriate bucket!!! If these are AEM sites (downloaded from the AEM AWS bucket), run the batch file “Results\_to\_AEM\_QA\_bucket” in the directory “C:\Matt-SFR Files\Hydraulic Modeling”. If these are CHaMP sites, run the batch file “Results\_to\_CHaMP\_QA\_bucket” from that same directory.

If you ran models from both the AEM and CHaMP buckets, delete all your results, go sit in a corner, and think about what you’ve done; then re-read step 5; then begin again, running AEM and CHaMP sites on different aws instances.

1. If running on an AWS instance, shut down and terminate AWS instance

Be sure to shut down and terminate the AWS instance upon completion! Failure to do so will result in continuous accrual of charges for using the instance, based on time, not processor usage. The instance does not shut down automatically.

**Appendix 1: Quality Assurance (QA) Procedure:**

A series of jpeg files are created for each model run. No models will have perfect boundary conditions, but the vast majority of models will have adequate boundary conditions. Some models can be improved by manual adjustment of boundary conditions.

The most important jpeg file to view is the boundary conditions image: **Boundary\_Condition.jpg**. The following provide examples of adequate boundary conditions and problematic boundary conditions where adjustments should be made:

Figure A1.1: d Boundary Conditions. Boundaries cut through all or most of the wetted width, and not too much of the site is cut off.

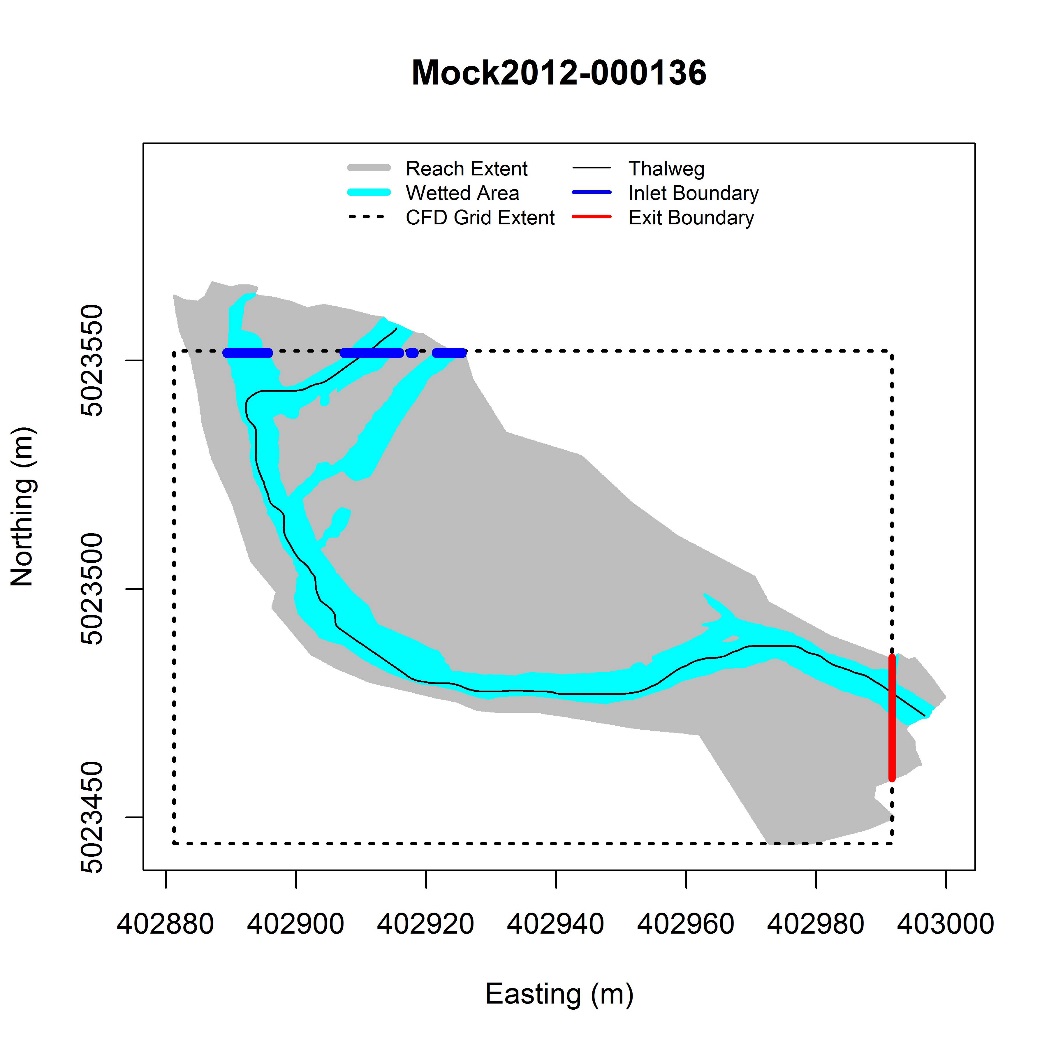


Figure A1.3: Trim length should be increased so that inlet boundary is distributed over both channels.

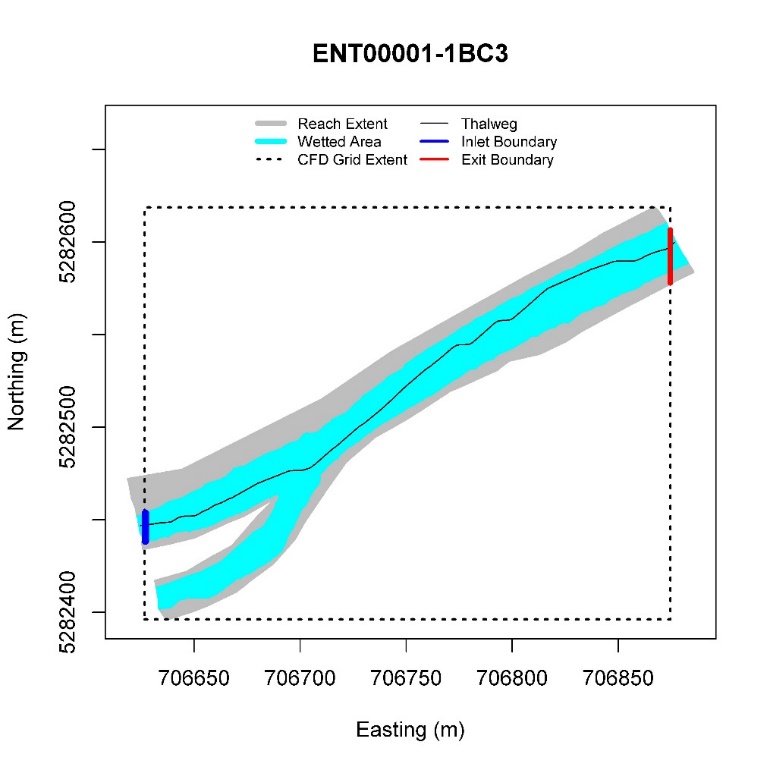


Figure A1.2: Good Boundary Conditions. Boundaries cut through all or most of the wetted width, and not too much of the site is cut off.

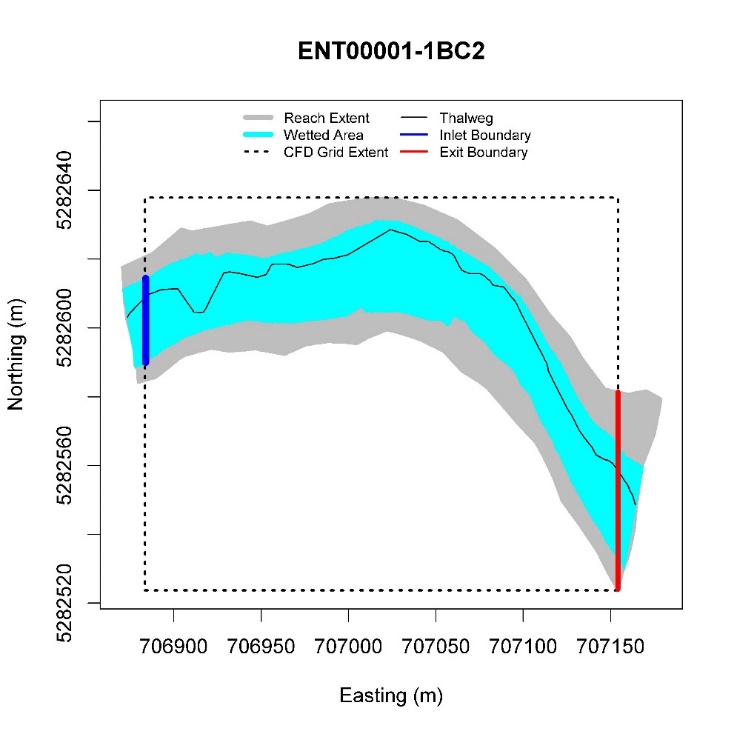


Figure A1.5B: Trim length Corrected

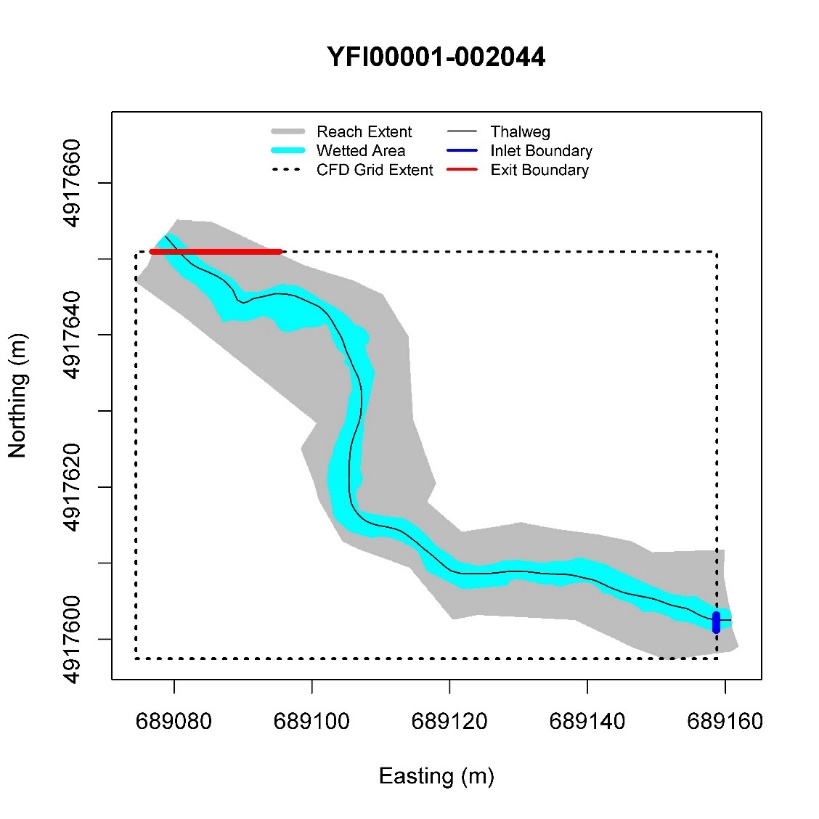


Figure A1.4B: Trim length reduced to achieve optimal boundary conditions.

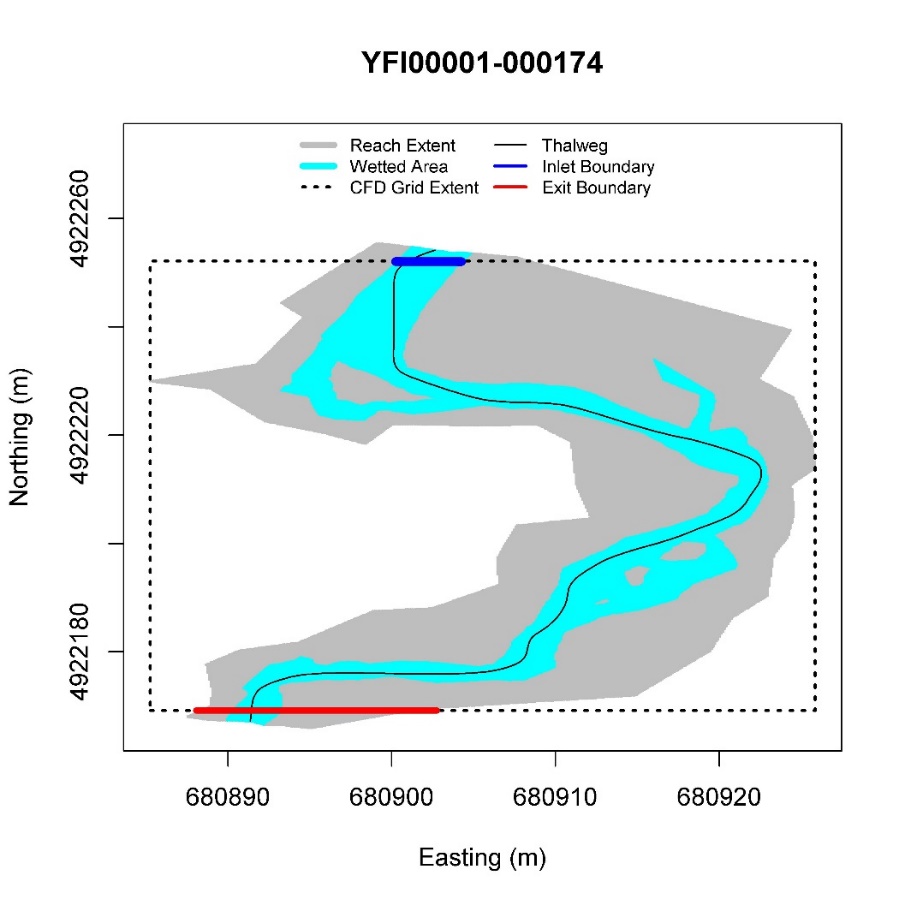


Figure A1.4A: Trim length should be decreased so that exit boundary cuts through channel at more ideal location

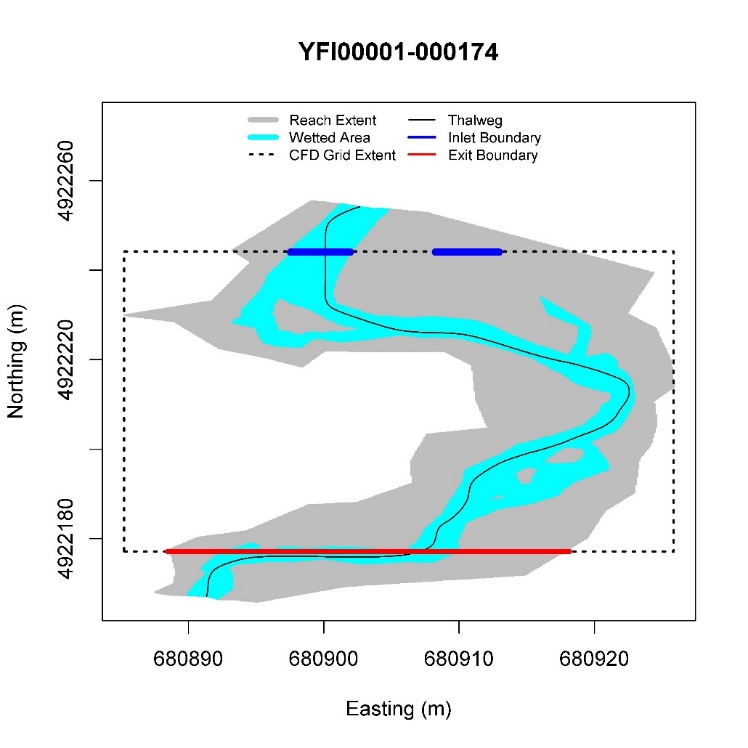


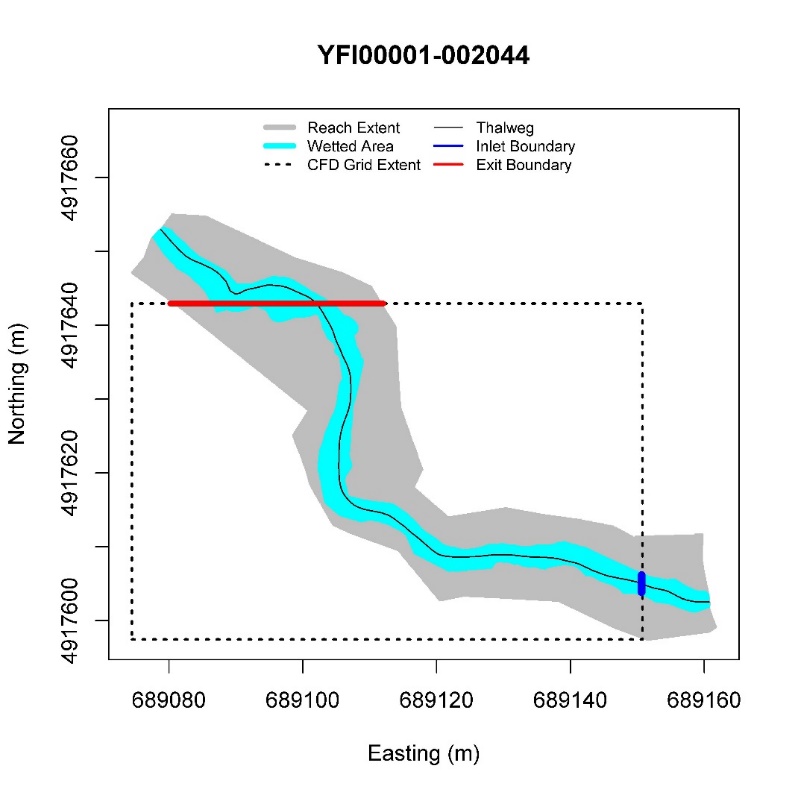
Figure A1.5A: Trim length should be decreased as too much of the site is being cut off

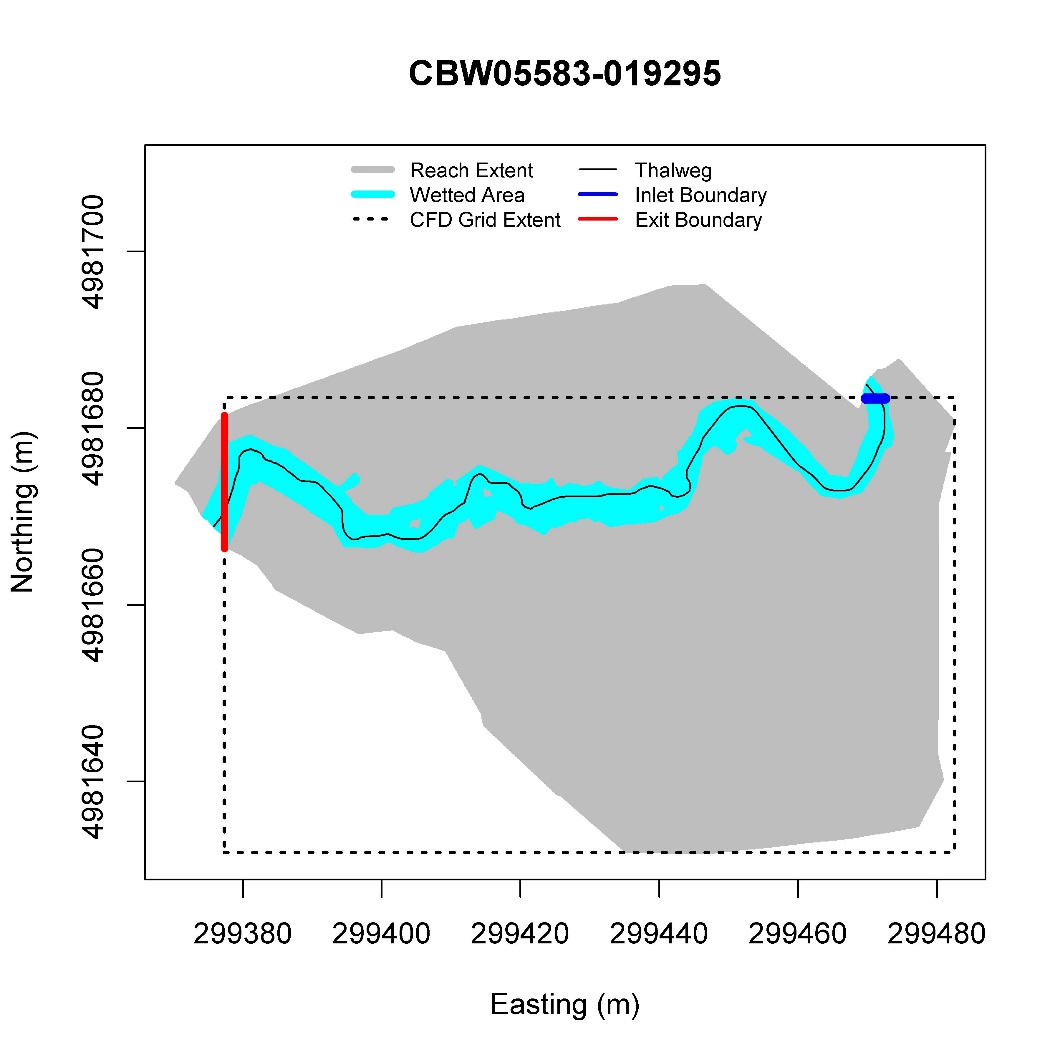
Figure A1.7B: Trim length issue corrected by reducing trim length.

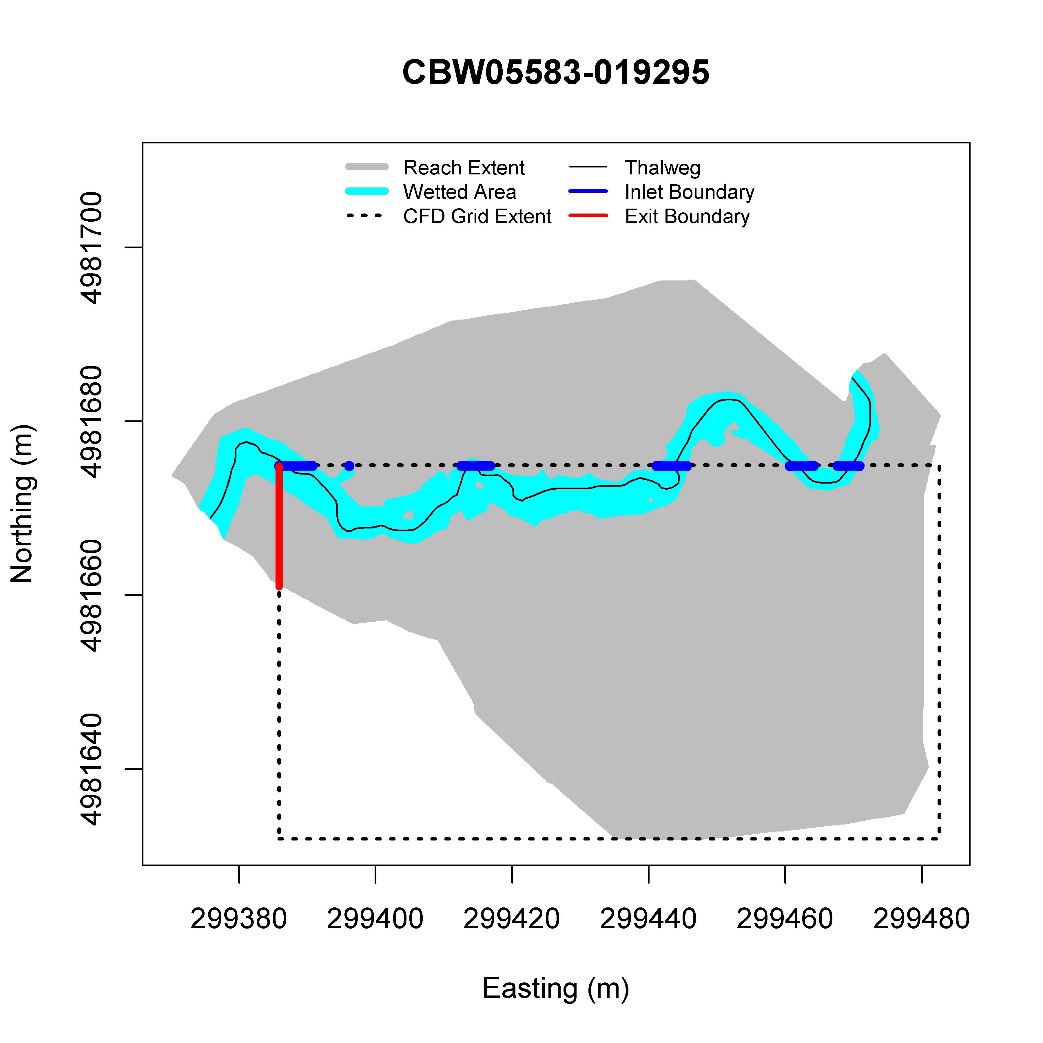
Figure A1.7A: Trim length issue. Reducing trim length should address issue. 

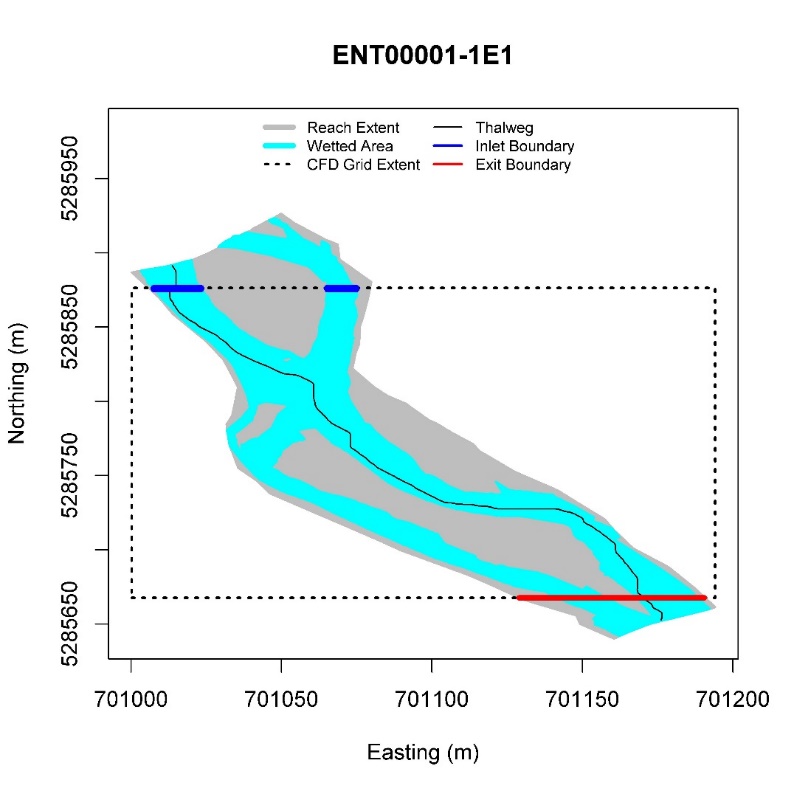
Figure A1.6B: Inlet boundary condition location corrected 

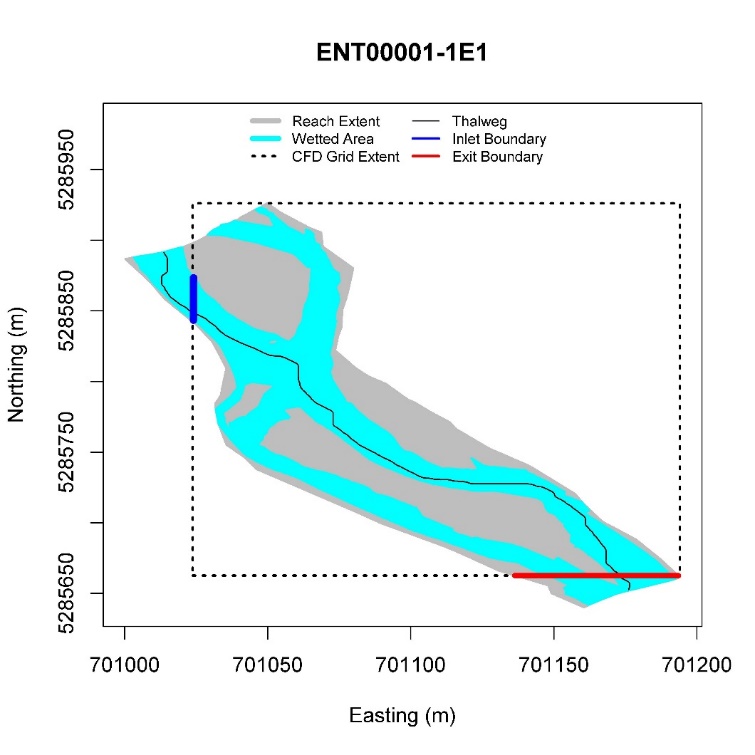
Figure A1.6A: Inlet boundary should be manually changed to the north boundary (see instructions on manual BC location override)

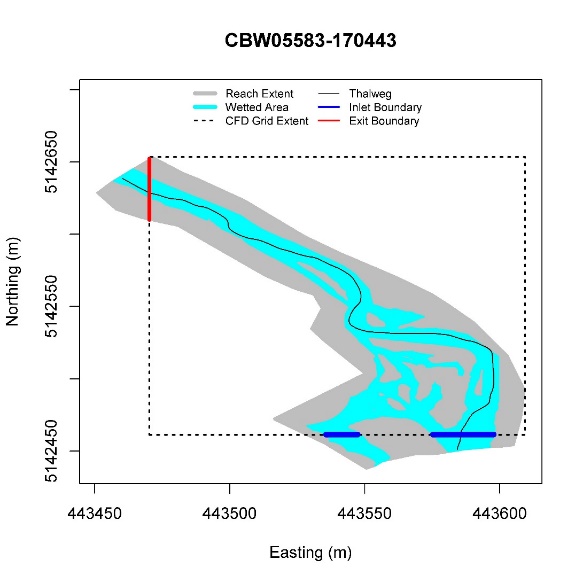
Figure A1.11: Adequate boundary condition. 

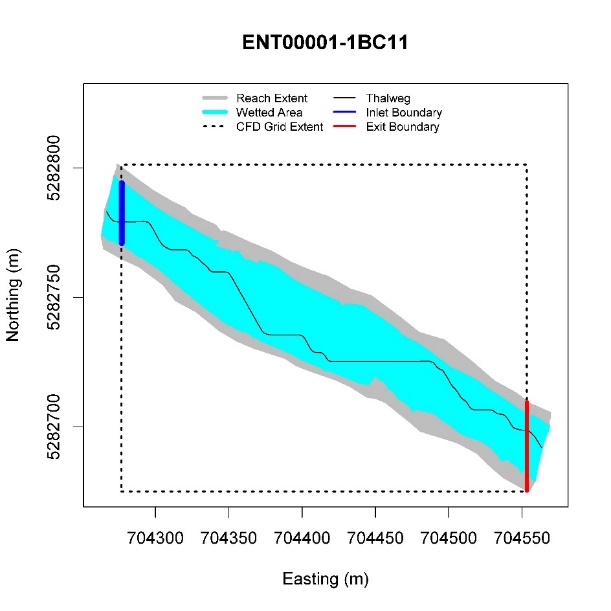
Figure A1.10: Adequate boundary condition. 

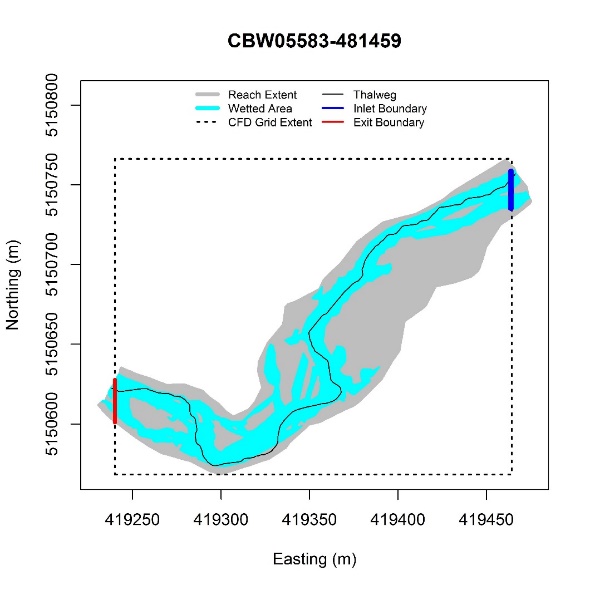
Figure A1.9: Adequate boundary condition. Note that the exit BC doesn’t quite cover the entire wetted width. Nonetheless, we generally consider this good enough. 

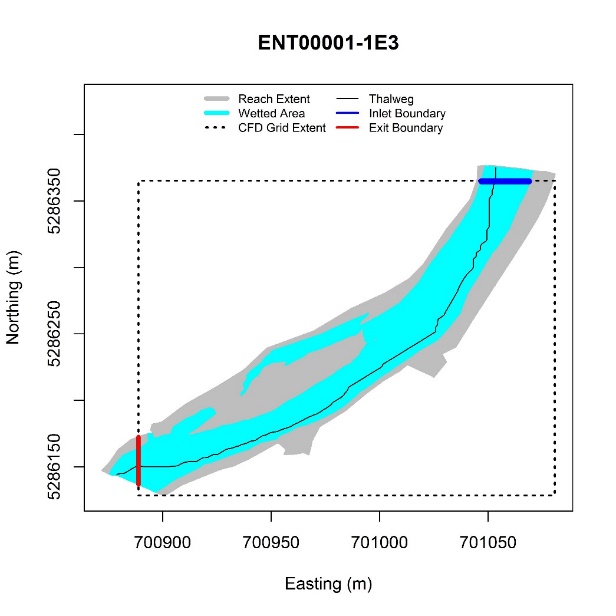
Figure A1.8: Adequate boundary condition. Note that the exit BC doesn’t quite cover the entire wetted width. Nonetheless, we generally consider this good enough. 

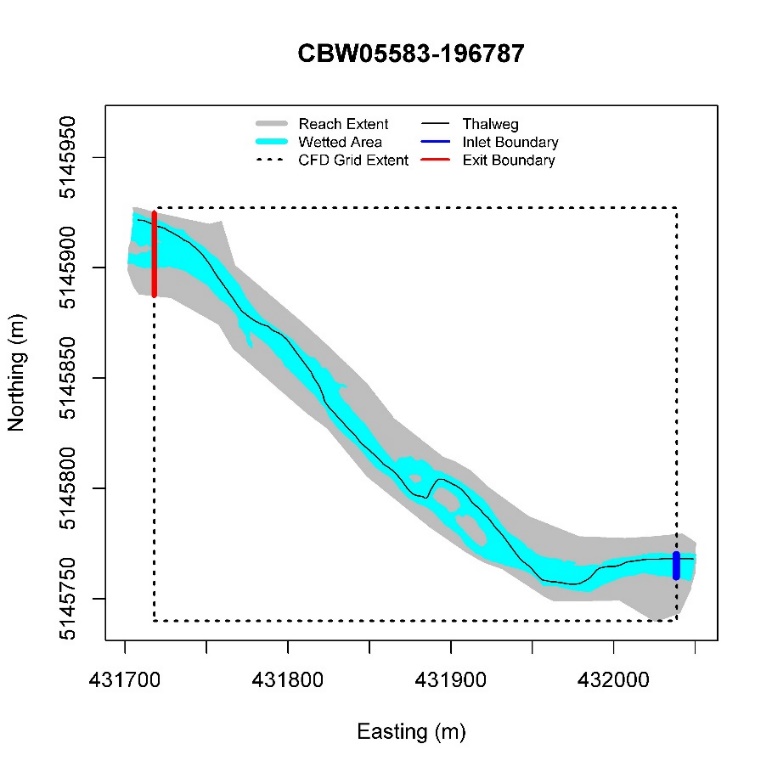
Figure A1.13B: Corrected by forcing the exit boundary to be on the west edge.

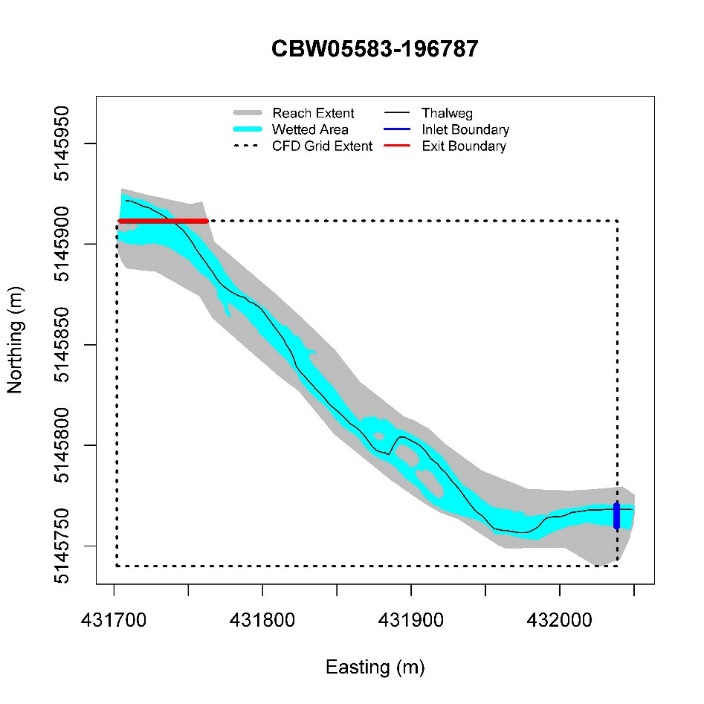
Figure A1.13A: This isn’t terrible, but could be improved by forcing the exit boundary to be on the west edge. 

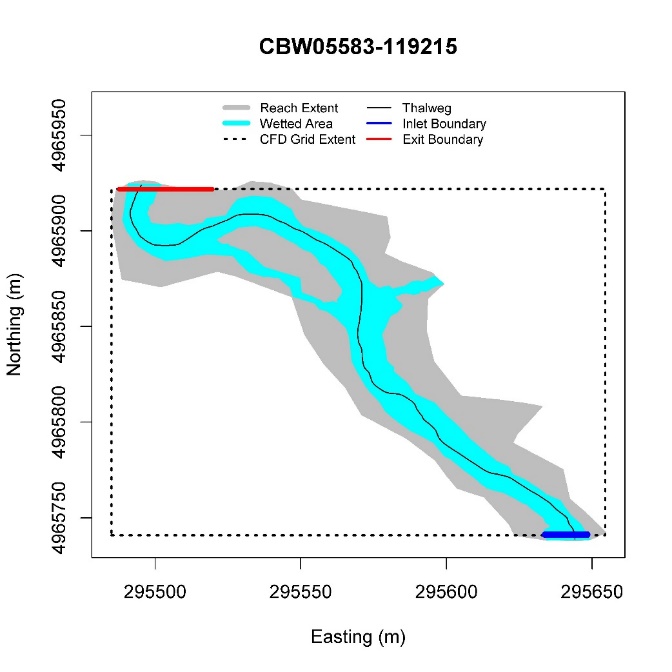
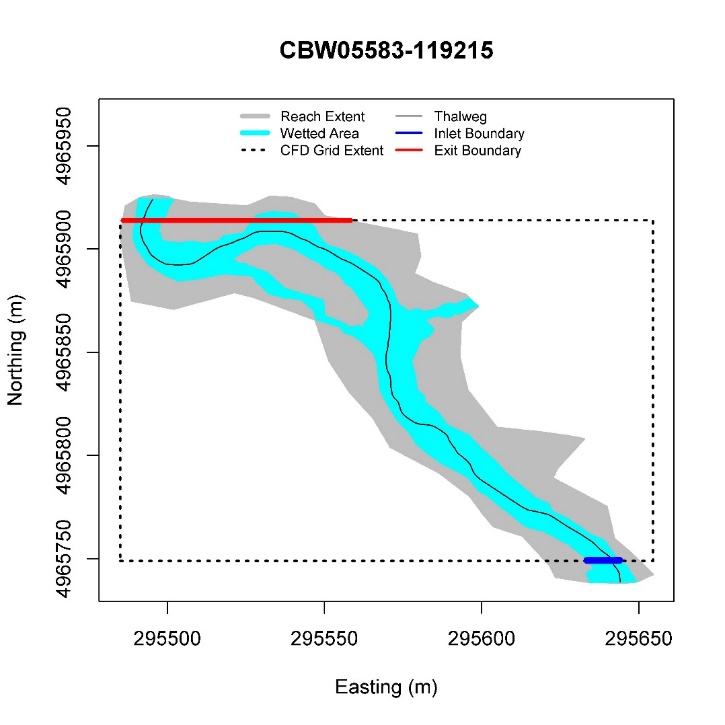
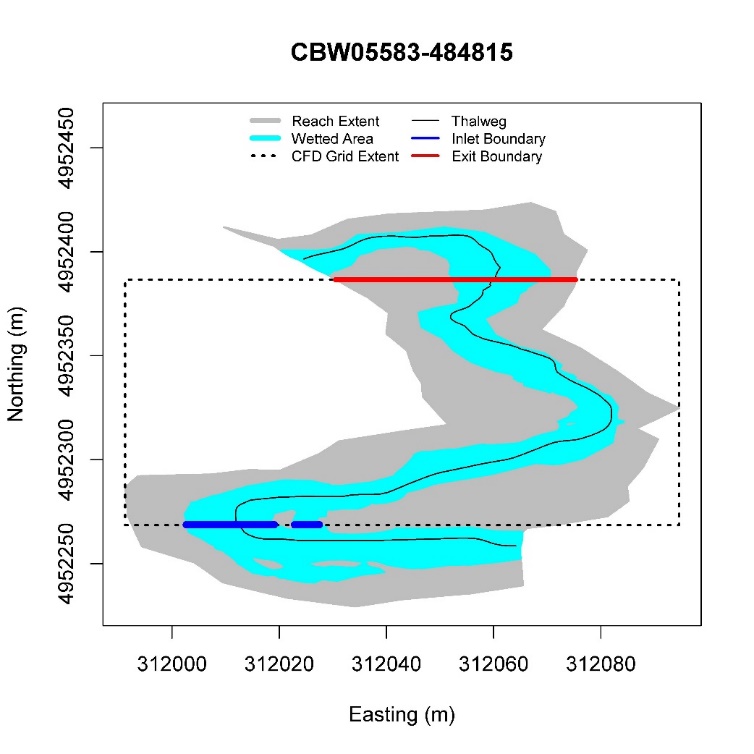
Figure A1.12B: Trim length corrected.

Figure A1.12A: Trim length should be reduced so exit BC doesn’t overlap non boundary portion of flow 

**Appendix 2: Porous Structure Modeling:**

Figure A1.14: No good solution available with our current process. This is what we accept – large trim length to ensure boundaries cut through channel width. 

Porous structure modeling can be done with the addition of a user input file and by using a different pre-processing R-code.

The input file must consist of three columns. The first two columns list the X and Y albers coordinates of every point at a site where the user wishes to specify porous structures. These porous structure coordinates must be on a 10 cm (.01 meter) grid. They shapes made up from the individual points may be as complicated as the user wishes, but it’s important to note that the structures will always be defined only as 2D structures. Water can flow though them (with some resistance), or around them, but not over or under them. The third column specifies the porosity, as a percent, in the range from zero to 100. Additional columns may exist the the right, but these columns will be ignored. The first three columns MUST be X, Y, and Porosity. See Table A.2.1

Table A.2.1: Porous plate input file example, showing the first 10 rows only

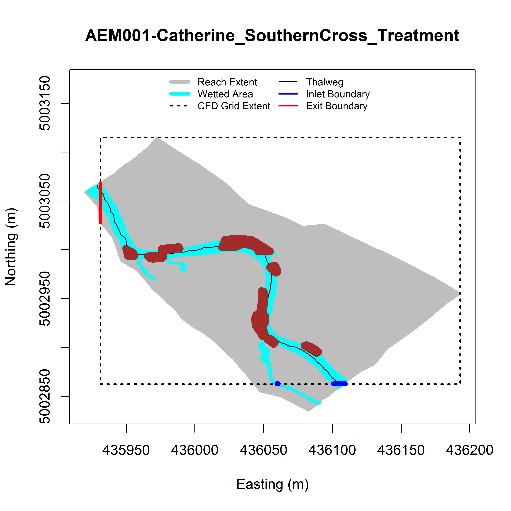


This file MUST be called “Jam\_Locations.csv”, and it must be located in the same directory as the DEM.csv, WSEDEM.csv, and thatlweg.csv files for each VisitID being modeled.

Note that any X-Y locations in “Jam\_Locations.csv” that are outside of the computational grid for a given hydro run will be ignored. (Andy Hill has used a “Jam\_Locations.csv” file that contains jams for multiple sites. The same “Jam\_Locations.csv” file is used for each site, but only those jam locations present in a site are included for that site, the rest are ignored. This saves the effort of having to break one large “Jam\_Locations.csv” file into multiple unique files for each site).

Once the “Jam\_Locations.csv” file is in the directory, follow the procedure as outlined in this document, except use the R file “Build\_Input\_Files\_Porous\_Plates.R” instead of the default “Build\_Input\_Files.R”. This alternate file performs all the functions of “Build\_Input\_Files.R” and the additional process steps necessary to model with porous plates.

If done correctly, you’ll have an additional QA file generated by the “Build\_Input\_Files\_Porous\_Plates.R” script that is not generated by the default code. This file is called “porous\_plates.jpg”. It is simply a copy of the boundary conditions plot with the location of the porous plates overlaid on top, so the user can confirm presence and location of the porous plates being modeled.

Figure A2.1. QA file showing porous plate locations overlaid onto boundary conditions QA plot. Porous plate locations are in maroon.

“Build\_Input\_Files\_Porous\_Plates.R” also generated an additional file in the input files to delft 3D called “test.ppl”, and creates a modified “test.mdf”that instructs Delft3D to read the porous plate input file and model using the porous structures. Details on these files and how the calculations are done can be found in the Delft 3D Flow documentation.

Other than adding the “Jam\_Locations.csv” file and using “Build\_Input\_Files\_Porous\_Plates.R”, porous plate modeling is done via exactly the same process as default modeling. Models may be run at non-default flow rates, D84 can be varied, etc.

If modeling using porous plates, please do NOT upload results to the sfr-champdata aws bucket or the sfr-aemdata aws bucket. At present, we have no way of tracking these results separately from default runs. Instead, simply zip up the results, name them appropriately and add documentation of what was modeled to the .zip folder, and deliver results directly to the customer. (I typically use Dropbox).