Based on your workflow and the WhiteboxTools documentation, here's a scientifically restructured version with proper sequencing and descriptive methods text:

**2.3 Hydrological Processing with WhiteboxTools**

1. **Methodological Considerations for Endorheic Watershed Delineation**

Lake Chilwa represents a unique hydrological challenge as an endorheic (closed) basin where water accumulates without surface outflow. Standard watershed delineation algorithms (O'Callaghan & Mark, 1984) assume exorheic drainage with defined pour points where water exits the system. For endorheic basins, we adapted the workflow by: (1) identifying terminal sink points at the lake's lowest elevations where maximum flow accumulation occurs, and (2) using the wbt\_basins() algorithm to delineate all sub-basins, from which the target watershed was extracted.

**Software and Data Requirements:**

* WhiteboxTools (v2.4.3+) via whitebox R package
* Spatial processing: terra, sf, stars packages
* SRTM 90m DEM resampled to 100m resolution (EPSG:3857)
* Lake boundary reference data

#| warning: false

#| message: false

#| error: false

#| eval: false

#| echo: true

#| comment: NA

# ============================================================================

# STEP 1: DEM HYDROLOGICAL CONDITIONING

# ============================================================================

# Objective: Remove artifacts while preserving the main endorheic depression

# 1a. Conservative depression breaching

# Algorithm: Breach shallow depressions (<3m) to remove DEM artifacts

# while preserving the primary Lake Chilwa basin

whitebox::wbt\_breach\_depressions(

dem = "dem\_chilwa\_00\_raw.tif",

output = "dem\_chilwa\_11\_breached.tif",

wd = "./assets/TIF/",

max\_depth = 3, # Only breach shallow artifacts

flat\_increment = 0.001

)

# 1b. Depression filling using Wang & Liu (2006) algorithm

# Optimized for flat terrain typical of marshlands

whitebox::wbt\_fill\_depressions\_wang\_and\_liu(

dem = "dem\_chilwa\_11\_breached.tif",

output = "dem\_chilwa\_12\_filled\_wang.tif",

wd = "./assets/TIF/"

)

# 1c. Quantify conditioning effects

dem\_breach <- terra::rast("./assets/TIF/dem\_chilwa\_11\_breached.tif")

dem\_filled <- terra::rast("./assets/TIF/dem\_chilwa\_12\_filled\_wang.tif")

depression\_effect <- dem\_filled - dem\_breach

terra::writeRaster(depression\_effect,

"./assets/TIF/dem\_chilwa\_13\_depression\_effects.tif",

overwrite = TRUE)

# ============================================================================

# STEP 2: FLOW DIRECTION AND ACCUMULATION

# ============================================================================

# Algorithm: D8 single-flow-direction (O'Callaghan & Mark, 1984)

# Note: D8 routes flow from each cell to its steepest downslope neighbor

# 2a. Calculate D8 flow direction

whitebox::wbt\_d8\_pointer(

dem = "dem\_chilwa\_12\_filled\_wang.tif",

output = "dem\_chilwa\_13\_flow\_direction\_D8.tif",

wd = "./assets/TIF/"

)

# 2b. Calculate flow accumulation

# Output units: number of upstream cells contributing flow

# At 100m resolution: 1 cell = 0.01 km² contributing area

whitebox::wbt\_d8\_flow\_accumulation(

input = "dem\_chilwa\_13\_flow\_direction\_D8.tif",

pntr = TRUE,

output = "dem\_chilwa\_14\_flow\_accumulation\_D8.tif",

wd = "./assets/TIF/"

)

# ============================================================================

# STEP 3: DEPRESSION NETWORK ANALYSIS

# ============================================================================

# Objective: Identify and characterize the endorheic depression system

# 3a. Calculate depth within depressions

# Uses raw DEM to measure depth below spillover points

whitebox::wbt\_depth\_in\_sink(

dem = "dem\_chilwa\_00\_raw.tif",

output = "dem\_chilwa\_15\_sink\_depth.tif",

wd = "./assets/TIF/",

zero\_background = FALSE

)

# 3b. Calculate topographic wetness index

# Indicates areas prone to water accumulation

whitebox::wbt\_wetness\_index(

sca = "dem\_chilwa\_14\_flow\_accumulation\_D8.tif",

slope = "dem\_chilwa\_12\_filled\_wang.tif",

output = "dem\_chilwa\_16\_wetness\_index.tif",

wd = "./assets/TIF/"

)

# ============================================================================

# STEP 4: TERMINAL POINT IDENTIFICATION

# ============================================================================

# Interactive selection of drainage terminus at lake center

# Points placed at maximum flow accumulation locations

flow\_accum <- terra::rast("./assets/TIF/dem\_chilwa\_14\_flow\_accumulation\_D8.tif")

outlets <- mapedit::editMap(mapview::mapview(flow\_accum))

outlets\_sf <- outlets$all |>

sf::st\_transform(crs\_master) |>

dplyr::select(geometry)

outlets\_sf$id <- "chilwa\_drainage\_terminus"

sf::st\_write(outlets\_sf, "./assets/SHP/outlets.shp",

delete\_layer = TRUE, quiet = TRUE)

# ============================================================================

# STEP 5: WATERSHED DELINEATION

# ============================================================================

# Strategy: Delineate all basins, then extract Lake Chilwa's watershed

# 5a. Delineate all drainage basins

# Algorithm: Basins (Lindsay, 2016) - identifies all complete basins

whitebox::wbt\_basins(

d8\_pntr = "dem\_chilwa\_13\_flow\_direction\_D8.tif",

output = "dem\_chilwa\_17\_basins.tif",

wd = "./assets/TIF/"

)

# 5b. Extract Lake Chilwa watershed by spatial intersection

all\_basins <- terra::rast("./assets/TIF/dem\_chilwa\_17\_basins.tif")

basins\_poly <- terra::as.polygons(all\_basins, dissolve = FALSE)

basins\_sf <- sf::st\_as\_sf(basins\_poly)

# Identify basin containing outlet point

outlet\_basin <- basins\_sf[sf::st\_intersects(basins\_sf, outlets\_sf, sparse = FALSE), ]

# Extract largest contiguous polygon

outlet\_basin$area\_km2 <- as.numeric(sf::st\_area(outlet\_basin)) / 1e6

chilwa\_watershed <- outlet\_basin[which.max(outlet\_basin$area\_km2), ]

chilwa\_watershed$id <- "lake\_chilwa\_basin"

sf::st\_write(chilwa\_watershed,

"./assets/SHP/chilwa\_watershed\_final.shp",

delete\_layer = TRUE)

# ============================================================================

# STEP 6: STREAM NETWORK EXTRACTION

# ============================================================================

# Objective: Delineate primary drainage channels

# 6a. Extract streams using flow accumulation threshold

# Threshold = 25 cells (0.25 km²) for flat marshland terrain

whitebox::wbt\_extract\_streams(

flow\_accum = "dem\_chilwa\_14\_flow\_accumulation\_D8.tif",

output = "dem\_chilwa\_18\_streams\_initial.tif",

threshold = 25,

zero\_background = TRUE,

wd = "./assets/TIF/"

)

# 6b. Remove short stream segments (<2 km)

whitebox::wbt\_remove\_short\_streams(

d8\_pntr = "dem\_chilwa\_13\_flow\_direction\_D8.tif",

streams = "dem\_chilwa\_18\_streams\_initial.tif",

output = "dem\_chilwa\_19\_streams\_cleaned.tif",

min\_length = 200, # 200 cells × 100m = 20 km

wd = "./assets/TIF/"

)

# 6c. Identify main stem channels

whitebox::wbt\_find\_main\_stem(

d8\_pntr = "dem\_chilwa\_13\_flow\_direction\_D8.tif",

streams = "dem\_chilwa\_19\_streams\_cleaned.tif",

output = "dem\_chilwa\_20\_streams\_trunk.tif",

wd = "./assets/TIF/"

)

# 6d. Convert raster streams to vector linestrings

whitebox::wbt\_raster\_streams\_to\_vector(

streams = "dem\_chilwa\_20\_streams\_trunk.tif",

d8\_pntr = "dem\_chilwa\_13\_flow\_direction\_D8.tif",

output = "streams\_chilwa\_network.shp",

wd = "./assets/TIF/"

)

# Load and calculate stream statistics

streams\_sf <- sf::st\_read("./assets/TIF/streams\_chilwa\_network.shp")

sf::st\_crs(streams\_sf) <- 3857

streams\_sf$length\_km <- as.numeric(sf::st\_length(streams\_sf)) / 1000

sf::st\_write(streams\_sf,

"./assets/SHP/chilwa\_streams\_final.shp",

delete\_layer = TRUE)

# ============================================================================

# SUMMARY STATISTICS

# ============================================================================

print(paste("Watershed area:", round(chilwa\_watershed$area\_km2, 1), "km²"))

print(paste("Total stream length:", round(sum(streams\_sf$length\_km), 1), "km"))

print(paste("Stream density:",

round(sum(streams\_sf$length\_km) / chilwa\_watershed$area\_km2, 2),

"km/km²"))

1. **References**

* Lindsay, J.B. (2016). Whitebox GAT: A case study in geomorphometric analysis. *Computers & Geosciences*, 95, 75-84.
* O'Callaghan, J.F., & Mark, D.M. (1984). The extraction of drainage networks from digital elevation data. *Computer Vision, Graphics, and Image Processing*, 28(3), 323-344.
* Wang, L., & Liu, H. (2006). An efficient method for identifying and filling surface depressions in digital elevation models. *International Journal of Geographical Information Science*, 20(2), 193-213.

This restructured version provides clear scientific documentation while maintaining computational efficiency.

# Load outputs

sink\_depth <- terra::rast("./assets/TIF/dem\_chilwa\_15\_sink\_depth.tif")

wetness\_idx <- terra::rast("./assets/TIF/dem\_chilwa\_16\_wetness\_index.tif")

flow\_accum <- terra::rast("./assets/TIF/dem\_chilwa\_14\_flow\_accumulation\_D8.tif")

# ============================================================================

# OPTION 1: Side-by-side comparison (recommended for reports)

# ============================================================================

tmap::tmap\_mode("plot")

# Stack for faceted display

depression\_stack <- c(sink\_depth, wetness\_idx)

names(depression\_stack) <- c("Sink Depth", "Wetness Index")

tmap::tm\_shape(depression\_stack) +

tmap::tm\_raster(

palette = "Blues",

style = "cont",

title = c("Depth (m)", "TWI")

) +

tmap::tm\_shape(lake) +

tmap::tm\_borders(col = "red", lwd = 2) +

tmap::tm\_facets(ncol = 2, free.scales = FALSE) +

tmap::tm\_layout(

panel.labels = c("A. Depth in Sink", "B. Topographic Wetness Index"),

panel.label.size = 1.2,

legend.position = c("left", "bottom"),

legend.text.size = 0.8

)

# ============================================================================

# OPTION 2: Classified thresholds (better for interpretation)

# ============================================================================

# Classify sink depth into meaningful categories

sink\_classes <- terra::classify(sink\_depth,

rcl = matrix(c(

0, 1, 1, # Shallow (<1m)

1, 5, 2, # Moderate (1-5m)

5, 10, 3, # Deep (5-10m)

10, 50, 4, # Very deep (10-50m)

50, Inf, 5 # Lake basin (>50m)

), ncol = 3, byrow = TRUE))

# Classify wetness index

wetness\_classes <- terra::classify(wetness\_idx,

rcl = matrix(c(

-Inf, 5, 1, # Dry

5, 10, 2, # Moderate

10, 15, 3, # Wet

15, 20, 4, # Very wet

20, Inf, 5 # Saturated

), ncol = 3, byrow = TRUE))

# Visualize classified

class\_stack <- c(sink\_classes, wetness\_classes)

names(class\_stack) <- c("Depression Class", "Wetness Class")

tmap::tm\_shape(class\_stack) +

tmap::tm\_raster(

palette = list("Blues", "YlGnBu"),

labels = list(

c("Shallow <1m", "Moderate 1-5m", "Deep 5-10m", "Very Deep 10-50m", "Lake >50m"),

c("Dry", "Moderate", "Wet", "Very Wet", "Saturated")

),

title = c("Depression Depth", "Wetness Level")

) +

tmap::tm\_shape(lake) +

tmap::tm\_borders(col = "black", lwd = 2) +

tmap::tm\_facets(ncol = 2) +

tmap::tm\_layout(

panel.labels = c("A. Depression Classification", "B. Wetness Classification"),

panel.label.size = 1.2,

legend.position = c("left", "bottom")

)

# ============================================================================

# OPTION 3: Interactive overlay (best for exploration)

# ============================================================================

tmap::tmap\_mode("view")

tmap::tm\_shape(sink\_depth) +

tmap::tm\_raster(

palette = "Blues",

alpha = 0.7,

title = "Sink Depth (m)",

style = "quantile",

n = 7

) +

tmap::tm\_shape(wetness\_idx) +

tmap::tm\_raster(

palette = "YlGnBu",

alpha = 0.5,

title = "Wetness Index",

style = "quantile",

n = 7

) +

tmap::tm\_shape(lake) +

tmap::tm\_borders(col = "red", lwd = 3) +

tmap::tm\_basemap("Esri.WorldImagery")

# ============================================================================

# OPTION 4: Overlay analysis (show relationship)

# ============================================================================

tmap::tmap\_mode("plot")

# Normalize both to 0-1 scale for comparison

sink\_norm <- (sink\_depth - min(values(sink\_depth), na.rm = TRUE)) /

(max(values(sink\_depth), na.rm = TRUE) - min(values(sink\_depth), na.rm = TRUE))

wetness\_norm <- (wetness\_idx - min(values(wetness\_idx), na.rm = TRUE)) /

(max(values(wetness\_idx), na.rm = TRUE) - min(values(wetness\_idx), na.rm = TRUE))

# Create RGB composite: Red = sink depth, Green = wetness

composite <- c(sink\_norm, wetness\_norm, sink\_norm \* 0)

tmap::tm\_shape(composite) +

tmap::tm\_rgb() +

tmap::tm\_shape(lake) +

tmap::tm\_borders(col = "white", lwd = 2) +

tmap::tm\_layout(

main.title = "Depression-Wetness Composite\n(Red = Depth, Green = Wetness)",

main.title.size = 1.2

)

# ============================================================================

# OPTION 5: Statistical summary visualization

# ============================================================================

# Extract values for histogram comparison

par(mfrow = c(1, 2))

hist(values(sink\_depth),

breaks = 50,

main = "Distribution of Sink Depths",

xlab = "Depth (m)",

col = "steelblue",

border = "white")

hist(values(wetness\_idx),

breaks = 50,

main = "Distribution of Wetness Index",

xlab = "TWI Value",

col = "darkgreen",

border = "white")

# ============================================================================

# OPTION 6: Zonal statistics by lake proximity

# ============================================================================

# Create distance buffer zones around lake

lake\_buffer <- terra::buffer(terra::vect(lake), width = c(5000, 10000, 20000))

lake\_zones <- terra::rasterize(lake\_buffer, sink\_depth, field = 1:3)

# Calculate mean depth and wetness by zone

zonal\_stats <- data.frame(

zone = c("0-5km", "5-10km", "10-20km", ">20km"),

mean\_depth = c(

mean(values(terra::mask(sink\_depth, lake\_zones == 1)), na.rm = TRUE),

mean(values(terra::mask(sink\_depth, lake\_zones == 2)), na.rm = TRUE),

mean(values(terra::mask(sink\_depth, lake\_zones == 3)), na.rm = TRUE),

mean(values(terra::mask(sink\_depth, is.na(lake\_zones))), na.rm = TRUE)

),

mean\_wetness = c(

mean(values(terra::mask(wetness\_idx, lake\_zones == 1)), na.rm = TRUE),

mean(values(terra::mask(wetness\_idx, lake\_zones == 2)), na.rm = TRUE),

mean(values(terra::mask(wetness\_idx, lake\_zones == 3)), na.rm = TRUE),

mean(values(terra::mask(wetness\_idx, is.na(lake\_zones))), na.rm = TRUE)

)

)

# Bar plot

barplot(zonal\_stats$mean\_depth,

names.arg = zonal\_stats$zone,

main = "Mean Depression Depth by Distance from Lake",

ylab = "Depth (m)",

col = "steelblue")