

# Package ‘manureshed’

December 22, 2025

**Type** Package

**Title** Spatiotemporal Nutrient Balance Analysis Across Agricultural and Municipal Systems

**Version** 0.1.0

**Maintainer** Olatunde D. Akanbi <olatunde.akanbi@case.edu>

**Description** A comprehensive framework for analyzing agricultural nutrient balances across multiple spatial scales (county, 'HUC8', 'HUC2') with integration of wastewater treatment plant ('WWTP') effluent loads for both nitrogen and phosphorus. Supports classification of spatial units as nutrient sources, sinks, or balanced areas based on agricultural surplus and deficit calculations. Includes visualization tools, spatial transition probability analysis, and nutrient flow network mapping. Built-in datasets include agricultural nutrient balance data from the Nutrient Use Geographic Information System ('NuGIS'; The Fertilizer Institute and Plant Nutrition Canada, 1987-2016) <[https://nugis.tfi.org/tabular\\_data/](https://nugis.tfi.org/tabular_data/)> and U.S. Environmental Protection Agency ('EPA') wastewater discharge data from the 'ECHO' Discharge Monitoring Report ('DMR') Loading Tool (2007-2016) <<https://echo.epa.gov/trends/loading-tool/water-pollution-search>>. Data are downloaded on demand from the Open Science Framework ('OSF') repository to minimize package size while maintaining full functionality. The integrated 'manureshed' framework methodology is described in Akanbi et al. (2025) <[doi:10.1016/j.resconrec.2025.108697](https://doi.org/10.1016/j.resconrec.2025.108697)>. Designed for nutrient management planning, environmental analysis, and circular economy research at watershed/administrative to national scales. This material is based upon financial support by the National Science Foundation, EEC Division of Engineering Education and Centers, NSF Engineering Research Center for Advancing Sustainable and Distributed Fertilizer Production (CASFER), NSF 20-553 Gen-4 Engineering Research Centers award 2133576. We thank Dr. Robert D. Sabo (U.S. Environmental Protection Agency) for his valuable contributions to the conceptual development and review of this work.

**License** MIT + file LICENSE

**URL** <https://osf.io/g39xa/>

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**Depends** R (>= 4.0.0)

**Imports** dplyr (>= 1.0.0), sf (>= 1.0.0), ggplot2 (>= 3.3.0), tidyr (>= 1.1.0), jsonlite (>= 1.7.0), rlang (>= 0.4.0), magrittr, scales (>= 1.1.0), igraph (>= 1.2.0), tigris (>= 1.5.0), stats, utils, tools

**Suggests** nhdplusTools (>= 0.5.0), RColorBrewer (>= 1.1.0), cowplot (>= 1.1.0), ggpubr (>= 0.4.0), viridis (>= 0.6.0), testthat (>= 3.0.0), knitr, progress, rmarkdown

**SystemRequirements** GDAL (>= 2.0.1), GEOS (>= 3.4.0), PROJ (>= 4.8.0)

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Olatunde D. Akanbi [aut, cre, cph] (ORCID: <<https://orcid.org/0000-0001-7719-2619>>),  
 Vibha Mandayam [aut] (ORCID: <<https://orcid.org/0009-0008-8628-9904>>),  
 Atharva Gupta [aut] (ORCID: <<https://orcid.org/0009-0004-5372-0260>>),  
 K. Colton Flynn [aut] (ORCID: <<https://orcid.org/0000-0002-5718-1071>>),  
 Jeffrey Yarus [aut] (ORCID: <<https://orcid.org/0000-0002-9331-9568>>),  
 Erika I. Barcelos [aut, cph] (ORCID: <<https://orcid.org/0000-0002-9273-8488>>),  
 Roger H. French [aut, cph] (ORCID: <<https://orcid.org/0000-0002-6162-0532>>)

**Repository** CRAN

**Date/Publication** 2025-12-22 18:10:02 UTC

## Contents

add_centroid_coordinates . . . . .	4
add_texas_huc8 . . . . .	4
agri_classify_complete . . . . .	5
agri_classify_complete_custom . . . . .	6
agri_classify_nitrogen . . . . .	7
agri_classify_nitrogen_custom . . . . .	7
agri_classify_phosphorus . . . . .	9
agri_classify_phosphorus_custom . . . . .	9
agri_process_nugis . . . . .	10
batch_analysis_enhanced . . . . .	11
batch_analysis_parallel . . . . .	12
batch_analysis_years . . . . .	13
benchmark_analysis . . . . .	14
calculate_cropland_threshold . . . . .	15
calculate_transition_probabilities . . . . .	16
check_builtin_data . . . . .	16
citation_info . . . . .	17
clean_category_names . . . . .	18

clean_text . . . . .	18
clear_data_cache . . . . .	19
compare_analyses . . . . .	19
CONUS_STATES . . . . .	20
convert_load_units . . . . .	20
create_analysis_report . . . . .	21
create_classification_summary . . . . .	22
create_network_plot . . . . .	22
data_sources . . . . .	23
download_all_data . . . . .	24
download_osf_data . . . . .	24
export_for_gis . . . . .	25
export_for_policy . . . . .	26
export_for_publication . . . . .	26
filter_by_state . . . . .	27
format_huc8 . . . . .	27
get_cropland_threshold . . . . .	28
get_nutrient_colors . . . . .	28
get_state_boundaries . . . . .	29
get_state_fips . . . . .	29
health_check . . . . .	30
integrate_complete . . . . .	30
integrate_wwtp_agricultural . . . . .	31
KG_TO_TONS . . . . .	32
LBS_TO_KG . . . . .	32
LBS_TO_TONS . . . . .	32
list_available_years . . . . .	33
load_built_in_boundaries . . . . .	33
load_built_in_nugis . . . . .	34
load_built_in_wwtp . . . . .	34
load_user_wwtp . . . . .	36
manureshed . . . . .	37
MANURESHEDED_CRIS . . . . .	42
map_agricultural_classification . . . . .	43
map_wwtp_columns . . . . .	43
map_wwtp_influence . . . . .	44
map_wwtp_points . . . . .	44
P2O5_TO_P . . . . .	45
plot_absolute_changes . . . . .	45
plot_before_after_comparison . . . . .	46
plot_impact_ratios . . . . .	46
quick_analysis . . . . .	47
quick_check . . . . .	48
quick_state_analysis . . . . .	49
run_built_in_analysis . . . . .	50
run_state_analysis . . . . .	52
save_analysis_summary . . . . .	53
save_centroid_data . . . . .	54

save_plot . . . . .	55
save_spatial_data . . . . .	56
save_transition_matrix . . . . .	57
summarize_results . . . . .	58
test_osf_connection . . . . .	60
validate_columns . . . . .	60
wwtp_aggregate_by_boundaries . . . . .	61
wwtp_classify_sources . . . . .	61
wwtp_clean_data . . . . .	62
wwtp_filter_positive_loads . . . . .	63
wwtp_process_complete . . . . .	63
wwtp_to_spatial . . . . .	64

<b>Index</b>	<b>66</b>
--------------	-----------

---

add\_centroid\_coordinates  
*Add Centroid Coordinates to Spatial Data*

---

### Description

Calculate centroid coordinates for spatial units

### Usage

```
add_centroid_coordinates(spatial_data)
```

### Arguments

spatial\_data sf object. Spatial data

### Value

Data frame with centroid coordinates added

---

add\_texas\_huc8 *Add Texas HUC8 Data (Updated for OSF)*

---

### Description

Add manually supplied Texas HUC8 data for missing watersheds Uses OSF data loading instead of built-in data

### Usage

```
add_texas_huc8(huc8_data, year = 2016, cropland_threshold, verbose = TRUE)
```

**Arguments**

huc8\_data        sf object. Existing HUC8 agricultural data  
 year             Numeric. Year to extract from Texas data  
 cropland\_threshold  
                   Numeric. Threshold for classification  
 verbose         Logical. Show progress messages

**Value**

sf object with Texas data added

---

agri\_classify\_complete

*Complete Agricultural Classification Pipeline*

---

**Description**

Run complete agricultural nutrient classification analysis for both N and P

**Usage**

```
agri_classify_complete(
  nugis_data,
  scale,
  cropland_threshold = NULL,
  county_data = NULL
)
```

**Arguments**

nugis\_data        Data frame. Raw NuGIS data  
 scale             Character. Spatial scale: "county", "huc8", or "huc2"  
 cropland\_threshold  
                   Numeric. Optional custom threshold  
 county\_data      Data frame. County data for threshold calculation (if needed)

**Value**

Data frame with complete agricultural classifications for both nutrients

---

`agri_classify_complete_custom`

*Complete Agricultural Classification Pipeline with Custom Efficiency Factors*

---

### Description

Run complete agricultural nutrient classification analysis for both N and P with user-specified efficiency factors for sensitivity analysis.

### Usage

```
agri_classify_complete_custom(  
  nugis_data,  
  scale,  
  cropland_threshold = NULL,  
  county_data = NULL,  
  n_efficiency = 0.5,  
  p_efficiency = 1  
)
```

### Arguments

<code>nugis_data</code>	Data frame. Raw NuGIS data
<code>scale</code>	Character. Spatial scale: "county", "huc8", or "huc2"
<code>cropland_threshold</code>	Numeric. Optional custom threshold
<code>county_data</code>	Data frame. County data for threshold calculation (if needed)
<code>n_efficiency</code>	Numeric. Nitrogen efficiency factor (default: 0.5)
<code>p_efficiency</code>	Numeric. Phosphorus efficiency factor (default: 1.0)

### Value

Data frame with complete agricultural classifications for both nutrients

### Examples

```
# Load county data  
nugis_data <- load_built_in_nugis("county", 2016)  
  
# Standard analysis  
results_standard <- agri_classify_complete_custom(  
  nugis_data, "county"  
)  
  
# Sensitivity analysis with varied nitrogen efficiency  
results_high_n <- agri_classify_complete_custom(
```

```
nugis_data, "county",
n_efficiency = 0.7
)

# Analysis with both custom efficiencies
results_custom <- agri_classify_complete_custom(
  nugis_data, "county",
  n_efficiency = 0.6,
  p_efficiency = 0.9
)
```

---

agri\_classify\_nitrogen

*Classify Agricultural Nitrogen Status*

---

### Description

Classify spatial units based on nitrogen balance using standard 0.5 efficiency factor

### Usage

```
agri_classify_nitrogen(data, cropland_threshold, scale = "huc8")
```

### Arguments

data	Data frame with processed agricultural data
cropland_threshold	Numeric. Threshold for excluding small cropland areas
scale	Character. Spatial scale for within-unit classification

### Value

Data frame with nitrogen classification added

---

agri\_classify\_nitrogen\_custom

*Classify Agricultural Nitrogen Status with Custom Efficiency Factor*

---

### Description

Classify spatial units based on nitrogen balance with user-specified efficiency factor. This function allows sensitivity analysis by varying the nitrogen efficiency assumption. The default value of 0.5 represents typical losses during nutrient cycling, uptake, and application, but regional conditions may warrant different values.

**Usage**

```
agri_classify_nitrogen_custom(
  data,
  cropland_threshold,
  scale = "huc8",
  n_efficiency = 0.5
)
```

**Arguments**

data	Data frame with processed agricultural data
cropland_threshold	Numeric. Threshold for excluding small cropland areas
scale	Character. Spatial scale for within-unit classification
n_efficiency	Numeric. Nitrogen efficiency factor (default: 0.5, range: 0-1)

**Value**

Data frame with nitrogen classification added

**Examples**

```
# Load and process data first
nugis_data <- load_built_in_nugis("county", 2016)
processed_data <- agri_process_nugis(nugis_data, "county")
cropland_threshold <- 500 * 2.47105 # 500 ha in acres

# Standard analysis with default 0.5 efficiency
results_default <- agri_classify_nitrogen_custom(
  processed_data, cropland_threshold = cropland_threshold, n_efficiency = 0.5
)

# Sensitivity analysis with higher efficiency (e.g., improved management)
results_high <- agri_classify_nitrogen_custom(
  processed_data, cropland_threshold = cropland_threshold, n_efficiency = 0.7
)

# Conservative analysis with lower efficiency
results_low <- agri_classify_nitrogen_custom(
  processed_data, cropland_threshold = cropland_threshold, n_efficiency = 0.3
)

# Compare classification changes across efficiency scenarios
table(results_default$N_class)
table(results_high$N_class)
table(results_low$N_class)
```



---

`agri_classify_phosphorus`*Classify Agricultural Phosphorus Status*

---

**Description**

Classify spatial units based on phosphorus balance (no efficiency factor for P)

**Usage**

```
agri_classify_phosphorus(data, cropland_threshold, scale = "huc8")
```

**Arguments**

<code>data</code>	Data frame with processed agricultural data
<code>cropland_threshold</code>	Numeric. Threshold for excluding small cropland areas
<code>scale</code>	Character. Spatial scale for within-unit classification

**Value**

Data frame with phosphorus classification added

---

`agri_classify_phosphorus_custom`*Classify Agricultural Phosphorus Status with Custom Efficiency Factor*

---

**Description**

Classify spatial units based on phosphorus balance with user-specified efficiency factor. While standard phosphorus classification uses 100% allows sensitivity analysis by varying the phosphorus efficiency assumption for different management scenarios or application methods.

**Usage**

```
agri_classify_phosphorus_custom(  
  data,  
  cropland_threshold,  
  scale = "huc8",  
  p_efficiency = 1  
)
```

**Arguments**

**data**                Data frame with processed agricultural data  
**cropland\_threshold**  
                       Numeric. Threshold for excluding small cropland areas  
**scale**                Character. Spatial scale for within-unit classification  
**p\_efficiency**        Numeric. Phosphorus efficiency factor (default: 1.0, range: 0-1)

**Value**

Data frame with phosphorus classification added

**Examples**

```

# Load and process data first
nugis_data <- load_built_in_nugis("county", 2016)
processed_data <- agri_process_nugis(nugis_data, "county")
cropland_threshold <- 500 * 2.47105 # 500 ha in acres

# Standard analysis with default 1.0 efficiency (100%)
results_default <- agri_classify_phosphorus_custom(
  processed_data, cropland_threshold = cropland_threshold, p_efficiency = 1.0
)

# Analysis with reduced efficiency (e.g., accounting for losses)
results_reduced <- agri_classify_phosphorus_custom(
  processed_data, cropland_threshold = cropland_threshold, p_efficiency = 0.8
)

# Conservative analysis with lower efficiency
results_conservative <- agri_classify_phosphorus_custom(
  processed_data, cropland_threshold = cropland_threshold, p_efficiency = 0.6
)

# Compare classification changes across efficiency scenarios
table(results_default$P_class)
table(results_reduced$P_class)
table(results_conservative$P_class)

```

---

agri\_process\_nugis      *Process NuGIS Data for Manureshed Analysis*

---

**Description**

Clean and standardize NuGIS data for agricultural nutrient analysis

**Usage**

```
agri_process_nugis(nugis_data, scale)
```

**Arguments**

nugis_data	Data frame. Raw NuGIS data for specified year
scale	Character. Spatial scale: "county", "huc8", or "huc2"

**Value**

Data frame with standardized columns for analysis

---

batch\_analysis\_enhanced

*Enhanced Batch Analysis with Full Visualizations*

---

**Description**

Run batch analysis with comprehensive visualization output for each year

**Usage**

```
batch_analysis_enhanced(
  years,
  scale = "huc8",
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = TRUE,
  output_base_dir = tempdir(),
  create_all_visualizations = TRUE,
  create_comparative_plots = TRUE,
  show_progress = TRUE,
  verbose = TRUE,
  ...
)
```

**Arguments**

years	Numeric vector. Years to analyze
scale	Character. Spatial scale
nutrients	Character vector. Nutrients to analyze
include_wwtp	Logical. Include WWTP analysis
output_base_dir	Character. Base output directory
create_all_visualizations	Logical. Create all maps, networks, and comparisons
create_comparative_plots	Logical. Create year-over-year comparisons
show_progress	Logical. Display progress bar (requires 'progress' package)
verbose	Logical. Show progress
...	Additional arguments

**Value**

List of results with comprehensive outputs

**Examples**

```
## Not run:  
# This function is computationally intensive  
# See vignette("advanced-features") for examples  
results <- batch_analysis_enhanced(years = 2015:2016)  
  
## End(Not run)
```

---

batch\_analysis\_parallel

*Batch Analysis with Parallel Processing*

---

**Description**

Run batch analysis using multiple cores for faster processing

**Usage**

```
batch_analysis_parallel(years, n_cores = NULL, ...)
```

**Arguments**

years	Numeric vector. Years to analyze
n_cores	Integer. Number of cores (default: detectCores() - 1)
...	Arguments passed to run_builtin_analysis

**Value**

List of results

**Examples**

```
results <- batch_analysis_parallel(  
  years = 2015:2016, # Use valid years only  
  n_cores = 2,      # Max 2 cores for CRAN  
  scale = "county", # Use county for faster processing  
  nutrients = "nitrogen"  
)
```

---

batch\_analysis\_years *Batch Analysis Across Multiple Years*

---

## Description

Run manureshed analysis across multiple years with consistent parameters

## Usage

```
batch_analysis_years(  
  years,  
  scale = "huc8",  
  nutrients = c("nitrogen", "phosphorus"),  
  include_wwtp = TRUE,  
  output_base_dir = tempdir(),  
  create_comparative_plots = TRUE,  
  verbose = TRUE,  
  ...  
)
```

## Arguments

years	Numeric vector. Years to analyze
scale	Character. Spatial scale: "county", "huc8", or "huc2"
nutrients	Character vector. Nutrients to analyze
include_wwtp	Logical. Whether to include WWTP (only available for 2007-2016 built-in)
output_base_dir	Character. Base output directory
create_comparative_plots	Logical. Whether to create year-over-year comparisons
verbose	Logical. Whether to print progress
...	Additional arguments passed to run_built_in_analysis

## Value

List of results for each year

## Examples

```
# Analyze trends with WWTP for subset of supported range  
batch_results <- batch_analysis_years(  
  years = 2010:2012, # Use smaller range for examples  
  scale = "huc8",  
  nutrients = "nitrogen",  
  include_wwtp = TRUE  
)
```

```

# Historical analysis without WWTP
historical_results <- batch_analysis_years(
  years = 1990:1992, # Use smaller range
  scale = "county",
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = FALSE
)

# Mixed analysis: some years with WWTP, some without
mixed_results <- batch_analysis_years(
  years = c(2005, 2010, 2015), # 2010,2015 will have WWTP
  scale = "huc8",
  nutrients = "nitrogen",
  include_wwtp = TRUE # Will only apply to 2010,2015
)

```

---

benchmark\_analysis      *Benchmark Analysis Performance*

---

### Description

Test analysis speed and memory usage

### Usage

```

benchmark_analysis(
  scale = "huc8",
  year = 2016,
  nutrients = "nitrogen",
  n_runs = 3,
  include_wwtp = TRUE
)

```

### Arguments

scale	Character. Spatial scale
year	Numeric. Year to test
nutrients	Character vector. Nutrients to analyze
n_runs	Integer. Number of benchmark runs (default: 3)
include_wwtp	Logical. Include WWTP processing

### Value

List with timing statistics and memory usage

**Examples**

```
# Benchmark HUC8 analysis - use smaller scale for faster testing
benchmark <- benchmark_analysis(
  scale = "county", # Use county for faster testing
  year = 2016,
  nutrients = "nitrogen",
  n_runs = 2 # Reduce runs for faster testing
)
print(benchmark)
```

---

```
calculate_cropland_threshold
```

*Calculate Cropland Threshold for Exclusion*

---

**Description**

Calculate cropland threshold for excluding small agricultural areas Uses county 500ha baseline to determine percentile for other scales

**Usage**

```
calculate_cropland_threshold(
  county_data,
  target_data,
  county_cropland_col,
  target_cropland_col,
  baseline_ha = 500
)
```

**Arguments**

county\_data      Data frame. County-level NuGIS data with cropland column  
target\_data      Data frame. Target scale data (HUC8, HUC2) with cropland column  
county\_cropland\_col  
                  Character. Name of cropland column in county data  
target\_cropland\_col  
                  Character. Name of cropland column in target data  
baseline\_ha      Numeric. Baseline cropland in hectares for exclusion (default: 500)

**Value**

Numeric. Threshold value for target scale

---

calculate\_transition\_probabilities  
*Calculate Spatial Transition Probabilities*

---

**Description**

Calculate transition probabilities between adjacent spatial units

**Usage**

```
calculate_transition_probabilities(  
  data,  
  class_column,  
  longitude_col = "longitude",  
  latitude_col = "latitude"  
)
```

**Arguments**

data	Data frame. Data with classification and coordinate columns
class_column	Character. Name of classification column
longitude_col	Character. Name of longitude column (default: "longitude")
latitude_col	Character. Name of latitude column (default: "latitude")

**Value**

Data frame with transition probabilities as percentages

---

check\_builtin\_data      *Check Data Availability from OSF*

---

**Description**

Check Data Availability from OSF

**Usage**

```
check_builtin_data(verbose = FALSE)
```

**Arguments**

verbose	Logical. Show detailed information about cache status
---------	---

**Value**

List showing available datasets and cache status



---

`citation_info`*Display Package Citation Information*

---

**Description**

Provides citation information for the package and data sources. Prints formatted citation text to the console for the manureshed package, the underlying research methodology paper (Akanbi et al., 2026), and the primary data sources (NuGIS agricultural data and EPA WWTP discharge data). The function is designed for users to easily obtain proper citations for publications and reports.

**Usage**

```
citation_info()
```

**Details**

This function takes no arguments. It prints citation information directly to the console using `message()` functions, which can be suppressed with `suppressMessages()` if needed.

**Value**

No return value, called for side effects. The function prints citation information to the console including:

- Package citation with version and OSF repository
- Research methodology paper citation
- NuGIS data source attribution
- EPA WWTP data source attribution
- Contact information for data sources

**Note**

This function requires no arguments and can be called simply as `citation_info()`.

**See Also**

[check\\_builtin\\_data](#) for data availability, [health\\_check](#) for package diagnostics

**Examples**

```
# Display citation information
citation_info()
```

---

clean\_category\_names    *Clean Category Names for Display*

---

**Description**

Clean Category Names for Display

**Usage**

```
clean_category_names(names)
```

**Arguments**

names                  Character vector of category names to clean

**Value**

Character vector of cleaned names

---

clean\_text              *Clean Text Data*

---

**Description**

Remove extra quotes and whitespace from text fields

**Usage**

```
clean_text(text)
```

**Arguments**

text                    Character vector to clean

**Value**

Character vector of cleaned text

---

clear_data_cache	<i>Clear Data Cache</i>
------------------	-------------------------

---

**Description**

Remove cached datasets to free up disk space

**Usage**

```
clear_data_cache(confirm = TRUE, verbose = TRUE)
```

**Arguments**

confirm	Logical. Require confirmation before deleting
verbose	Logical. Show what's being deleted

**Value**

Logical. TRUE if successful

---

compare_analyses	<i>Compare Two Analysis Results</i>
------------------	-------------------------------------

---

**Description**

Compare classifications between two analysis results

**Usage**

```
compare_analyses(results1, results2, nutrient = "nitrogen")
```

**Arguments**

results1	First analysis results
results2	Second analysis results
nutrient	Character. Nutrient to compare ("nitrogen" or "phosphorus")

**Value**

Data frame with comparison

**Examples**

```
results_2010 <- run_builtin_analysis(scale = "county", year = 2010)
results_2016 <- run_builtin_analysis(scale = "county", year = 2016)
comparison <- compare_analyses(results_2010, results_2016, "nitrogen")
```

---

CONUS_STATES	<i>CONUS States</i>
--------------	---------------------

---

**Description**

Vector of Continental United States state abbreviations

**Usage**

```
CONUS_STATES
```

**Format**

An object of class character of length 49.

---

convert_load_units	<i>Convert Load Units to Tons</i>
--------------------	-----------------------------------

---

**Description**

Convert loads from various units to US tons

**Usage**

```
convert_load_units(load_values, from_unit)
```

**Arguments**

load_values	Numeric vector of load values
from_unit	Character. Input unit: "kg", "lbs", "pounds", "tons"

**Value**

Numeric vector of loads in US tons

**Examples**

```
# Convert from kilograms to tons
kg_loads <- c(1000, 2000, 3000)
tons_loads <- convert_load_units(kg_loads, "kg")

# Convert from pounds to tons
lbs_loads <- c(5000, 10000, 15000)
tons_loads <- convert_load_units(lbs_loads, "lbs")
```

---

```
create_analysis_report
```

*Create Analysis Report*

---

## Description

Generate comprehensive HTML or PDF report of analysis results

## Usage

```
create_analysis_report(  
  results,  
  output_path,  
  format = "html",  
  title = "Manureshed Analysis Report",  
  include_maps = TRUE  
)
```

## Arguments

results	List. Complete analysis results
output_path	Character. Path for output report
format	Character. Report format: "html" or "pdf"
title	Character. Report title
include_maps	Logical. Whether to include maps in report

## Value

Character. Path to generated report

## Examples

```
# Generate HTML report - use tempdir to avoid check directory pollution  
results <- run_builtin_analysis(scale = "county", year = 2016)  
report_path <- file.path(tempdir(), "analysis_report.html")  
create_analysis_report(results, report_path)
```

---

`create_classification_summary`*Create Classification Summary Table*

---

**Description**

Create summary table of classification counts for both nutrients

**Usage**

```
create_classification_summary(data, agricultural_col, combined_col)
```

**Arguments**

<code>data</code>	Data frame. Data with classification columns
<code>agricultural_col</code>	Character. Name of agricultural classification column
<code>combined_col</code>	Character. Name of combined (with WWTP) classification column

**Value**

Data frame with classification counts and changes

---

`create_network_plot`*Create Network Plot from Transition Probabilities*

---

**Description**

Create network visualization of spatial transition probabilities

**Usage**

```
create_network_plot(  
  transition_df,  
  nutrient,  
  analysis_type,  
  output_path,  
  highlight_transitions = TRUE  
)
```

**Arguments**

transition_df	Data frame. Transition probability matrix
nutrient	Character. "nitrogen" or "phosphorus" for coloring
analysis_type	Character. Description of analysis type
output_path	Character. Path to save PNG file
highlight_transitions	Logical. Whether to highlight specific transitions

**Value**

NULL (saves plot to file)

---

data_sources	<i>NuGIS and EPA WWTP Data Sources</i>
--------------	--

---

**Description**

This package uses two primary data sources:

**NuGIS Agricultural Data**

The Nutrient Use Geographic Information System (NuGIS) presents cropland nutrient balances for the conterminous United States from 1987-2016.

**Source:** The Fertilizer Institute (TFI) and Plant Nutrition Canada (PNC)

**Website:** [https://nugis.tfi.org/tabular\\_data](https://nugis.tfi.org/tabular_data)

**Contact:** [nugis@tfi.org](mailto:nugis@tfi.org)

**Components:**

- County-level crop and livestock data from USDA Census of Agriculture
- Fertilizer use data from AAPFCO
- Geospatial nutrient balance estimates
- Available for counties, HUC8, and HUC2 watersheds

**Data Processing:** The manureshed package uses cleaned versions of NuGIS data with resolved metadata issues and enhanced spatial integration, as detailed in the manureshed methodology paper (Akanbi et al., 2026).

**EPA WWTP Data**

**Source:** U.S. Environmental Protection Agency

**System:** Discharge Monitoring Report (DMR) Loading Tool via ECHO

**Website:** <https://echo.epa.gov/trends/loading-tool/water-pollution-search>

**Data Years:** 2007-2016 (nitrogen and phosphorus loads)

**License:** Public domain (U.S. Government work)

**Data Attribution**

When using this package, please cite both the package, methodology paper, and the underlying data sources. Use `citation_info()` to display full citation information.

---

download\_all\_data      *Download All Datasets from OSF*

---

**Description**

Convenience function to download all available datasets from OSF

**Usage**

```
download_all_data(force_download = FALSE, verbose = TRUE)
```

**Arguments**

force\_download Logical. Re-download even if files exist in cache  
 verbose Logical. Show progress for each download

**Value**

Logical. TRUE if all downloads successful

---

download\_osf\_data      *Download and Cache Data from OSF*

---

**Description**

Download built-in datasets from OSF repository using Files API

**Usage**

```
download_osf_data(  
  dataset_name,  
  force_download = FALSE,  
  cache_dir = NULL,  
  verbose = TRUE  
)
```

**Arguments**

dataset\_name Character. Name of dataset to download  
 force\_download Logical. Force re-download even if cached version exists  
 cache\_dir Character. Directory to cache downloaded data (default: user data dir)  
 verbose Logical. Show download progress



**Value**

Path to cached data file

---

export\_for\_gis      *Export Results for GIS Applications*

---

**Description**

Export spatial results in GIS-ready formats

**Usage**

```
export_for_gis(  
  results,  
  output_dir = file.path(tempdir(), "gis_export"),  
  formats = c("shapefile", "geojson")  
)
```

**Arguments**

results	Analysis results object
output_dir	Output directory
formats	Character vector of formats: "shapefile", "geojson", "kml", "gpkg"

**Value**

List of created files

**Examples**

```
# Use tempdir to avoid polluting check directory  
results <- run_builtin_analysis(scale = "county", year = 2016)  
output_dir <- file.path(tempdir(), "gis_outputs")  
gis_files <- export_for_gis(results, output_dir)
```

---

export\_for\_policy      *Export for Policy Briefs*

---

**Description**

Create simplified outputs for policy makers

**Usage**

```
export_for_policy(results, output_dir = file.path(tempdir(), "policy_export"))
```

**Arguments**

results	Analysis results
output_dir	Output directory

**Value**

List of created files

---

export\_for\_publication  
*Export for Publication*

---

**Description**

Create high-resolution outputs suitable for publication

**Usage**

```
export_for_publication(  
  results,  
  output_dir = file.path(tempdir(), "publication_export"),  
  dpi = 600  
)
```

**Arguments**

results	Analysis results
output_dir	Output directory
dpi	Resolution (default: 600 for publication quality)

**Value**

List of created files

---

filter_by_state	<i>Filter Data by State</i>
-----------------	-----------------------------

---

**Description**

Filter spatial data to a specific state

**Usage**

```
filter_by_state(data, state, scale, boundaries = NULL)
```

**Arguments**

data	Data frame or sf object. Spatial data with FIPS or HUC codes
state	Character. Two-letter state abbreviation
scale	Character. Spatial scale: "county", "huc8", or "huc2"
boundaries	sf object. Spatial boundaries (optional, for HUC scales)

**Value**

Filtered data for the specified state

---

format_huc8	<i>Format HUC8 Codes</i>
-------------	--------------------------

---

**Description**

Add leading zeros to 7-digit HUC8 codes to make them 8-digit

**Usage**

```
format_huc8(huc_codes)
```

**Arguments**

huc_codes	Character or numeric vector of HUC codes
-----------	--

**Value**

Character vector of properly formatted 8-digit HUC codes

---

`get_cropland_threshold`*Get Cropland Threshold by Scale*

---

**Description**

Get appropriate cropland threshold based on spatial scale

**Usage**

```
get_cropland_threshold(  
  scale,  
  county_data = NULL,  
  target_data = NULL,  
  baseline_ha = 500  
)
```

**Arguments**

<code>scale</code>	Character. Spatial scale: "county", "huc8", or "huc2"
<code>county_data</code>	Data frame. County-level data (required for huc8/huc2)
<code>target_data</code>	Data frame. Target scale data (required for huc8/huc2)
<code>baseline_ha</code>	Numeric. Baseline for county exclusion (default: 500)

**Value**

Numeric. Threshold value

---

`get_nutrient_colors`*Default Color Schemes for Nutrient Classifications*

---

**Description**

Default Color Schemes for Nutrient Classifications

**Usage**

```
get_nutrient_colors(nutrient = "nitrogen")
```

**Arguments**

<code>nutrient</code>	Character. Either "nitrogen" or "phosphorus"
-----------------------	--

**Value**

Named vector of colors for classification categories

---

get\_state\_boundaries    *Get State Boundaries for Mapping*

---

**Description**

Get US state boundaries excluding non-CONUS states

**Usage**

get\_state\_boundaries()

**Value**

sf object with state boundaries

---

get\_state\_fips    *Get State FIPS Code*

---

**Description**

Convert state abbreviation to FIPS code

**Usage**

get\_state\_fips(state\_abbr)

**Arguments**

state\_abbr    Character. Two-letter state abbreviation (e.g., "OH", "TX")

**Value**

Character. Two-digit state FIPS code

---

health_check	<i>Check Package Health and Dependencies</i>
--------------	--

---

**Description**

Diagnostic function to check package installation and dependencies

**Usage**

```
health_check(verbose = FALSE)
```

**Arguments**

verbose            Logical. Whether to display detailed information

**Value**

Logical. TRUE if all checks pass

**Examples**

```
# Quick health check
health_check()

# Detailed diagnostic information
health_check(verbose = TRUE)
```

---

integrate_complete	<i>Complete Integration Pipeline (Updated)</i>
--------------------	--

---

**Description**

Run complete integration of WWTP and agricultural data for both nutrients

**Usage**

```
integrate_complete(
  agri_data,
  wwtp_nitrogen_aggregated,
  wwtp_phosphorus_aggregated,
  cropland_threshold,
  scale = "huc8",
  add_texas = FALSE,
  year = 2016
)
```

**Arguments**

agri\_data           Data frame or sf object. Agricultural classification data  
 wwtp\_nitrogen\_aggregated           Data frame. Aggregated nitrogen WWTP data  
 wwtp\_phosphorus\_aggregated           Data frame. Aggregated phosphorus WWTP data  
 cropland\_threshold           Numeric. Threshold for exclusion  
 scale                Character. Spatial scale  
 add\_texas           Logical. Whether to add Texas HUC8 data (only for HUC8 scale)  
 year                 Numeric. Year for Texas data

**Value**

List with integrated nitrogen and phosphorus data

---

integrate\_wwtp\_agricultural  
*Integrate WWTP Data with Agricultural Classifications (Updated)*

---

**Description**

Combine WWTP loads with agricultural nutrient balance classifications

**Usage**

```

integrate_wwtp_agricultural(
  agri_data,
  wwtp_aggregated,
  nutrient,
  cropland_threshold,
  scale = "huc8"
)

```

**Arguments**

agri\_data           Data frame. Agricultural classification data  
 wwtp\_aggregated           Data frame. Aggregated WWTP loads by spatial unit  
 nutrient            Character. "nitrogen" or "phosphorus"  
 cropland\_threshold           Numeric. Threshold for exclusion classification  
 scale                Character. Spatial scale for within-unit classification

**Value**

Data frame with combined WWTP + agricultural classifications

---

KG\_TO\_TONS                      *Conversion Factor: Kilograms to US Tons*

---

**Description**

Conversion Factor: Kilograms to US Tons

**Usage**

KG\_TO\_TONS

**Format**

An object of class numeric of length 1.

---

LBS\_TO\_KG                      *Conversion Factor: Pounds to Kilograms*

---

**Description**

Conversion Factor: Pounds to Kilograms

**Usage**

LBS\_TO\_KG

**Format**

An object of class numeric of length 1.

---

LBS\_TO\_TONS                      *Conversion Factor: Pounds to US Tons*

---

**Description**

Conversion Factor: Pounds to US Tons

**Usage**

LBS\_TO\_TONS

**Format**

An object of class numeric of length 1.



---

list\_available\_years *List Available Built-in Years*

---

**Description**

Show available years for each data type

**Usage**

```
list_available_years(scale = NULL)
```

**Arguments**

scale            Character. Spatial scale (optional)

**Value**

Data frame with available years by data type

---

load\_builtin\_boundaries  
*Load Built-in Spatial Boundaries from OSF*

---

**Description**

Load built-in spatial boundary data for specified scale from OSF repository

**Usage**

```
load_builtin_boundaries(scale, force_download = FALSE, verbose = TRUE)
```

**Arguments**

scale            Character. Spatial scale: "county", "huc8", or "huc2"  
force\_download Logical. Force re-download even if cached  
verbose         Logical. Show download progress

**Value**

sf object with spatial boundaries

---

load\_builtin\_nugis      *Load Built-in NuGIS Data from OSF*

---

### Description

Load built-in NuGIS data from OSF repository for specified year and spatial scale. Data includes all years from 1987 through 2016.

### Usage

```
load_builtin_nugis(scale, year = 2016, force_download = FALSE, verbose = TRUE)
```

### Arguments

scale	Character. Spatial scale: "county", "huc8", or "huc2"
year	Numeric. Year to filter data (available: 1987-2016)
force_download	Logical. Force re-download even if cached
verbose	Logical. Show download progress

### Value

Data frame of NuGIS data for specified scale and year

### Examples

```
# Load county data for 2016
county_2016 <- load_builtin_nugis("county", 2016)

# Load HUC8 data for 2010
huc8_2010 <- load_builtin_nugis("huc8", 2010)

# Load county data for 2010, force fresh download
county_2010 <- load_builtin_nugis("county", 2010, force_download = TRUE)
```

---

load\_builtin\_wwtp      *Load Built-in WWTP Data from OSF*

---

### Description

Load built-in WWTP data for specified year from OSF repository (2007-2016 available)  
 Load built-in WWTP data for specified year from OSF repository (2007-2016 available)

**Usage**

```
load_builtin_wwtp(  
  nutrient,  
  year = 2016,  
  force_download = FALSE,  
  verbose = TRUE  
)
```

```
load_builtin_wwtp(  
  nutrient,  
  year = 2016,  
  force_download = FALSE,  
  verbose = TRUE  
)
```

**Arguments**

nutrient	Character. "nitrogen" or "phosphorus"
year	Numeric. Year to filter data (available: 2007-2016)
force_download	Logical. Force re-download even if cached
verbose	Logical. Show download progress

**Value**

Data frame with cleaned WWTP data for specified year

Data frame with cleaned WWTP data for specified year

**Examples**

```
# Load WWTP data for different years (2007-2016 available)  
wwtp_n_2016 <- load_builtin_wwtp("nitrogen", 2016)  
wwtp_n_2012 <- load_builtin_wwtp("nitrogen", 2012)  
wwtp_n_2007 <- load_builtin_wwtp("nitrogen", 2007)  
  
# Load phosphorus data  
wwtp_p_2015 <- load_builtin_wwtp("phosphorus", 2015)  
wwtp_p_2010 <- load_builtin_wwtp("phosphorus", 2010)  
  
# Force re-download  
wwtp_fresh <- load_builtin_wwtp("nitrogen", 2014, force_download = TRUE)
```

---

load_user_wwtp	<i>Load User WWTP Data</i>
----------------	----------------------------

---

### Description

Load and standardize user-provided WWTP data with flexible formatting

Load and standardize user-provided WWTP data with flexible formatting

### Usage

```
load_user_wwtp(  
  file_path,  
  nutrient,  
  column_mapping = NULL,  
  skip_rows = 0,  
  header_row = 1,  
  load_units = "kg"  
)
```

```
load_user_wwtp(  
  file_path,  
  nutrient,  
  column_mapping = NULL,  
  skip_rows = 0,  
  header_row = 1,  
  load_units = "kg"  
)
```

### Arguments

file_path	Character. Path to WWTP data file
nutrient	Character. "nitrogen" or "phosphorus"
column_mapping	Named list. Custom column mapping (optional)
skip_rows	Numeric. Number of rows to skip (default: 0)
header_row	Numeric. Row containing headers (default: 1)
load_units	Character. Units of pollutant loads: "kg", "lbs", "pounds", "tons"

### Value

Data frame with standardized WWTP data

Data frame with standardized WWTP data

## Examples

```
## Not run:
# Standard EPA format but will not run because data needs to be supplied as custom
# Load custom WWTP data (for years outside 2007-2016)
wwtp_data <- load_user_wwtp("nitrogen_2020.csv", "nitrogen")

# For years 2007-2016, consider using built-in data:
# wwtp_builtin <- load_builtin_wwtp("nitrogen", 2015)

# Custom format with different units
wwtp_data <- load_user_wwtp("custom_wwtp.csv", "phosphorus",
                           load_units = "lbs", skip_rows = 3)

# Custom column mapping
custom_map <- list(facility_name = "Plant_Name",
                  pollutant_load = "Load_lbs_per_year")
wwtp_data <- load_user_wwtp("custom.csv", "nitrogen", custom_map)

## End(Not run)
## Not run:

# Standard EPA format
wwtp_data <- load_user_wwtp("nitrogen_2020.csv", "nitrogen")

# Custom format with different units
wwtp_data <- load_user_wwtp("custom_wwtp.csv", "phosphorus",
                           load_units = "lbs", skip_rows = 3)

# Custom column mapping
custom_map <- list(facility_name = "Plant_Name",
                  pollutant_load = "Load_lbs_per_year")
wwtp_data <- load_user_wwtp("custom.csv", "nitrogen", custom_map)

## End(Not run)
```

---

manureshed

*manureshed: Manureshed Analysis with WWTP Integration*

---

## Description

This package provides comprehensive tools for analyzing agricultural nutrient balances at multiple spatial scales with optional integration of wastewater treatment plant (WWTP) nutrient loads for both nitrogen and phosphorus.

## Details

All datasets are downloaded on-demand from OSF repository to minimize package size while maintaining full functionality.

## Main Functions

### Data Loading and Management::

- `load_builtin_nugis`: Load NuGIS data from OSF
- `load_builtin_boundaries`: Load spatial boundaries from OSF
- `load_builtin_wwtp`: Load WWTP data from OSF (2016)
- `check_builtin_data`: Check available datasets and cache status
- `download_all_data`: Download all datasets at once
- `download_osf_data`: Download specific dataset
- `clear_data_cache`: Clear cached datasets

### High-Level Workflows::

- `run_builtin_analysis`: Complete end-to-end analysis workflow
- `quick_analysis`: Analysis with automatic visualizations
- `batch_analysis_years`: Multi-year analysis workflow

### Agricultural Classification::

- `agri_process_nugis`: Process and standardize NuGIS data
- `agri_classify_nitrogen`: Classify nitrogen balance
- `agri_classify_phosphorus`: Classify phosphorus balance
- `agri_classify_complete`: Complete agricultural pipeline

### WWTP Processing::

- `load_user_wwtp`: Load custom WWTP data with flexible formatting
- `wwtp_clean_data`: Clean and filter WWTP data
- `wwtp_classify_sources`: Classify WWTP facilities by load size
- `convert_load_units`: Handle different load units (kg, lbs, tons)

### Data Integration::

- `integrate_wwtp_agricultural`: Combine WWTP and agricultural data
- `integrate_complete`: Complete integration pipeline
- `add_texas_huc8`: Add Texas HUC8 supplemental data

### Visualization and Mapping::

- `map_agricultural_classification`: Map nutrient classifications
- `map_wwtp_points`: Map WWTP facility locations
- `map_wwtp_influence`: Map WWTP influence/proportion
- `get_state_boundaries`: Get US state boundaries for mapping

### Spatial Analysis::

- `calculate_transition_probabilities`: Spatial transition analysis
- `create_network_plot`: Network visualization of transitions
- `add_centroid_coordinates`: Calculate spatial centroids

### Comparison Analysis::

- `create_classification_summary`: Before/after comparison summaries
- `plot_before_after_comparison`: Comparison bar plots
- `plot_impact_ratios`: Impact ratio visualizations
- `plot_absolute_changes`: Absolute change plots

#### Utility Functions::

- `get_nutrient_colors`: Get color schemes for nutrients
- `clean_category_names`: Clean classification names for display
- `format_huc8`: Format HUC8 codes with leading zeros
- `get_cropland_threshold`: Calculate exclusion thresholds

### Spatial Scales

The package supports analysis at three spatial scales:

- **County**: US county boundaries (3,000+ units)
- **HUC8**: 8-digit Hydrologic Unit Code watersheds (2,000+ units)
- **HUC2**: 2-digit Hydrologic Unit Code regions (18 units)

### Nutrients Supported

The package supports analysis for both major nutrients with appropriate methodologies:

- **Nitrogen**: Uses 0.5 availability factor for manure nitrogen in calculations
- **Phosphorus**: Direct calculation without availability factor

Users can analyze one nutrient, both nutrients, or different combinations in the same workflow:  
`nutrients = c("nitrogen", "phosphorus")`

### Classification System

Spatial units are classified into five categories based on nutrient balance:

- **Source**: Net nutrient surplus available for export
- **Sink Deficit**: Total deficit requiring nutrient imports
- **Sink Fertilizer**: Fertilizer surplus available for manure import
- **Within Watershed/County**: Balanced for internal nutrient transfers
- **Excluded**: Insufficient cropland for meaningful analysis (<500 ha equivalent)

### Data Sources

The package provides access to comprehensive built-in datasets:

#### NuGIS Data::

- County-level data (1987 - 2016)
- HUC8 watershed data (1987 - 2016)
- HUC2 regional data (1987 - 2016)

- All nutrient balance components (manure, fertilizer, removal, fixation)

**Spatial Boundaries::**

- US county boundaries (CONUS)
- HUC8 watershed boundaries
- HUC2 regional boundaries
- All in Albers Equal Area Conic projection (EPSG:5070)

**WWTP Data::**

- Nitrogen discharge data (2007 - 2016)
- Phosphorus discharge data (2007 - 2016)
- Pre-processed and classification-ready
- Includes facility metadata and spatial coordinates

**Supplemental Data::**

- Texas HUC8 data (automatically integrated for HUC8 analyses)
- Texas spatial boundaries

**OSF Data Repository**

All datasets are hosted on OSF and downloaded on-demand:

- **Repository:** <https://osf.io/g39xa/>
- **Automatic caching:** Data downloaded once and reused
- **Flexible loading:** Load only the data you need
- **Version control:** Permanent DOI for reproducibility
- **Size efficiency:** Package <1MB, full datasets ~25MB

**WWTP Data Flexibility**

The package handles varying EPA WWTP data formats across different years:

- **Unit Conversion:** Automatic conversion between kg, lbs, pounds, and tons
- **Column Mapping:** Flexible mapping to handle EPA naming changes
- **Header Detection:** Support for different header row positions
- **Built-in 2016:** Ready-to-use cleaned data for immediate analysis
- **Custom Integration:** Easy integration of user data for other years

**Workflow Examples**

```
# Check what data is available
check_builtin_data()

# Download all datasets (optional, ~40MB)
download_all_data()
```



```
# Basic analysis using built-in data - any year 2007-2016
results <- run_builtin_analysis(
  scale = "huc8",
  year = 2012, # Any year 2007-2016 now supported
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = TRUE
)

wwtp_n_2010 <- load_builtin_wwtp("nitrogen", year = 2010)
wwtp_p_2015 <- load_builtin_wwtp("phosphorus", year = 2015)

# Quick analysis with automatic visualizations
viz_results <- quick_analysis(
  scale = "county",
  year = 2016,
  nutrients = "nitrogen",
  include_wwtp = TRUE
)

# Historical analysis without WWTP
historical <- run_builtin_analysis(
  scale = "huc8",
  year = 2010,
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = FALSE
)

# Load specific datasets manually
county_2016 <- load_builtin_nugis("county", 2016)
boundaries <- load_builtin_boundaries("county")
wwtp_n <- load_builtin_wwtp("nitrogen")
```

### Analysis Outputs

The package generates comprehensive outputs for each nutrient analyzed:

- **Spatial Data:** Classification results as sf objects
- **Maps:** Classification, influence, and facility maps
- **Networks:** Transition probability visualizations
- **Comparisons:** Before/after WWTP integration analysis
- **Data Files:** CSV centroids, RDS spatial data
- **Metadata:** Analysis parameters and processing information

### Performance and Scalability

- Optimized for CONUS-scale analysis (1000s of spatial units)
- Memory-efficient spatial operations

- On-demand data loading reduces memory footprint
- Progress reporting for long-running analyses
- Automatic garbage collection for memory management

**Author(s)**

**Maintainer:** Olatunde D. Akanbi <olatunde.akanbi@case.edu> ([ORCID](#)) [copyright holder]

Authors:

- Vibha Mandayam ([ORCID](#))
- Atharva Gupta ([ORCID](#))
- K. Colton Flynn ([ORCID](#))
- Jeffrey Yarus ([ORCID](#))
- Erika I. Barcelos <erika.barcelos@case.edu> ([ORCID](#)) [copyright holder]
- Roger H. French <roger.french@case.edu> ([ORCID](#)) [copyright holder]

**See Also**

Useful links:

- <https://osf.io/g39xa/>

---

MANURESHED\_CRS

*Standard CRS for Manureshed Analysis*

---

**Description**

Albers Equal Area Conic projection (EPSG:5070)

**Usage**

MANURESHED\_CRS

**Format**

An object of class `numeric` of length 1.

---

map\_agricultural\_classification  
*Create Agricultural Classification Map*

---

**Description**

Create map showing agricultural nutrient classifications

**Usage**

```
map_agricultural_classification(data, nutrient, classification_col, title)
```

**Arguments**

data	sf object. Spatial data with classifications
nutrient	Character. "nitrogen" or "phosphorus"
classification_col	Character. Name of classification column
title	Character. Map title

**Value**

ggplot object

---

map\_wwtp\_columns      *Map WWTP Column Names to Standard Format*

---

**Description**

Create mapping between EPA WWTP data column names and standardized format. Handles various EPA data formats across different years.

**Usage**

```
map_wwtp_columns(raw_data, nutrient, custom_mapping = NULL)
```

**Arguments**

raw_data	Data frame. Raw WWTP data
nutrient	Character. "nitrogen" or "phosphorus"
custom_mapping	Named list. Custom column mappings (optional)

**Value**

Named list with column mappings

## Examples

```
## Not run:
mapping <- map_wwtp_columns(raw_wwtp_data, "nitrogen")

# Custom mapping for different format
custom_map <- list(facility_name = "Plant_Name",
                  pollutant_load = "Annual_Load_kg")
mapping <- map_wwtp_columns(raw_data, "nitrogen", custom_map)

## End(Not run)
```

---

map\_wwtp\_influence      *Create WWTP Influence Map*

---

## Description

Create map showing WWTP contribution as proportion of total nutrient load

## Usage

```
map_wwtp_influence(data, nutrient, title)
```

## Arguments

data	sf object. Integrated data with WWTP proportions
nutrient	Character. "nitrogen" or "phosphorus"
title	Character. Map title

## Value

ggplot object

---

map\_wwtp\_points      *Create WWTP Point Map*

---

## Description

Create map showing WWTP locations classified by load size

## Usage

```
map_wwtp_points(wwtp_sf, nutrient, title)
```

**Arguments**

`wwtp_sf` sf object. Spatial WWTP data with classifications  
`nutrient` Character. "nitrogen" or "phosphorus"  
`title` Character. Map title

**Value**

ggplot object

---

P2O5\_TO\_P                      *Conversion Factor: P2O5 to P*

---

**Description**

Conversion Factor: P2O5 to P

**Usage**

P2O5\_TO\_P

**Format**

An object of class `numeric` of length 1.

---

`plot_absolute_changes`    *Create Absolute Change Plot*

---

**Description**

Create plot showing absolute changes in classification counts

**Usage**

`plot_absolute_changes(data, title)`

**Arguments**

`data`                      Data frame. Summary data with absolute changes  
`title`                      Character. Plot title

**Value**

ggplot object

---

plot\_before\_after\_comparison  
*Create Before/After Comparison Plot*

---

**Description**

Create side-by-side comparison of agricultural vs WWTP+agricultural classifications

**Usage**

```
plot_before_after_comparison(data, nutrient, title)
```

**Arguments**

data	Data frame. Summary data from create_classification_summary
nutrient	Character. "nitrogen" or "phosphorus" for coloring
title	Character. Plot title

**Value**

ggplot object

---

plot\_impact\_ratios     *Create Impact Ratio Plot*

---

**Description**

Create plot showing impact of WWTP addition as ratios

**Usage**

```
plot_impact_ratios(data, title)
```

**Arguments**

data	Data frame. Summary data with impact ratios
title	Character. Plot title

**Value**

ggplot object

---

quick_analysis	<i>Quick Analysis with Visualization</i>
----------------	--

---

## Description

Run analysis and automatically generate key visualizations for specified nutrients. This is a convenience function that combines `run_builtin_analysis` with automatic visualization generation.

## Usage

```
quick_analysis(
  scale = "huc8",
  year = 2016,
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = TRUE,
  output_dir = tempdir(),
  create_maps = TRUE,
  create_networks = TRUE,
  create_comparisons = TRUE,
  create_wwtp_maps = TRUE,
  wwtp_load_units = "kg",
  map_resolution = "medium",
  generate_report = FALSE,
  verbose = TRUE,
  ...
)
```

## Arguments

<code>scale</code>	Character. Spatial scale: "county", "huc8", or "huc2"
<code>year</code>	Numeric. Year to analyze
<code>nutrients</code>	Character vector. Nutrients to analyze: <code>c("nitrogen", "phosphorus")</code> or subset
<code>include_wwtp</code>	Logical. Whether to include WWTP analysis (default: TRUE)
<code>output_dir</code>	Character. Output directory (default: <code>tempdir()</code> )
<code>create_maps</code>	Logical. Whether to create classification maps (default: TRUE)
<code>create_networks</code>	Logical. Whether to create network plots (default: TRUE)
<code>create_comparisons</code>	Logical. Whether to create comparison plots (default: TRUE)
<code>create_wwtp_maps</code>	Logical. Whether to create WWTP facility maps (default: TRUE)
<code>wwtp_load_units</code>	Character. Units for WWTP loads if using custom data (default: "kg")
<code>map_resolution</code>	Character. Map resolution: "low", "medium", "high" (default: "medium")

```

generate_report Logical. Whether to generate HTML report (default: FALSE)
verbose         Logical. Whether to print progress messages (default: TRUE)
...            Additional arguments passed to run_built_in_analysis

```

**Value**

List with results and file paths of created visualizations

**Examples**

```

# Quick analysis with all visualizations (2007-2016 WWTP available)
results <- quick_analysis(
  scale = "huc8",
  year = 2012, # Use valid year
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = TRUE,
  generate_report = TRUE
)

# Agricultural only analysis for pre-WWTP year
results <- quick_analysis(
  scale = "county",
  year = 2005, # Before WWTP data
  nutrients = "nitrogen",
  include_wwtp = FALSE,
  create_networks = FALSE
)

# High-resolution analysis with expanded year range
results <- quick_analysis(
  scale = "huc8",
  year = 2008, # Use valid WWTP year
  nutrients = "phosphorus",
  include_wwtp = TRUE,
  map_resolution = "high"
)

```

---

quick\_check

*Quick Data Check*

---

**Description**

Perform quick validation checks on analysis results

**Usage**

```
quick_check(results, verbose = TRUE)
```



**Arguments**

results	Analysis results object
verbose	Logical. Print detailed messages

**Value**

Logical. TRUE if all checks pass

---

quick\_state\_analysis *Quick State Analysis with Visualization*

---

**Description**

Run state-level analysis with automatic visualizations

**Usage**

```
quick_state_analysis(
  state,
  scale = "huc8",
  year = 2016,
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = TRUE,
  output_dir = file.path(tempdir(), paste0("state_", tolower(state), "_results")),
  create_maps = TRUE,
  create_networks = TRUE,
  create_comparisons = TRUE,
  verbose = TRUE,
  ...
)
```

**Arguments**

state	Character. Two-letter state abbreviation
scale	Character. Spatial scale
year	Numeric. Year to analyze
nutrients	Character vector. Nutrients to analyze
include_wwtp	Logical. Include WWTP analysis
output_dir	Character. Output directory
create_maps	Logical. Create maps
create_networks	Logical. Create network plots
create_comparisons	Logical. Create comparison plots
verbose	Logical. Show progress
...	Additional arguments

**Value**

List with results and visualizations

**Examples**

```
# Quick state analysis - use states with good data coverage
results <- quick_state_analysis(
  state = "TX", # Texas has good data coverage
  scale = "county",
  year = 2016,
  nutrients = "nitrogen",
  include_wwtp = TRUE
)
```

---

run\_builtin\_analysis *Complete Manureshed Analysis Workflow (Built-in Data)*

---

**Description**

Run complete manureshed analysis using built-in NuGIS data (start-2016) and optional WWTP data. For WWTP analysis beyond 2016, users must provide their own data. Supports analysis of nitrogen, phosphorus, or both nutrients simultaneously.

**Usage**

```
run_builtin_analysis(
  scale = "huc8",
  year = 2016,
  nutrients = c("nitrogen", "phosphorus"),
  output_dir = tempdir(),
  include_wwtp = TRUE,
  wwtp_year = NULL,
  custom_wwtp_nitrogen = NULL,
  custom_wwtp_phosphorus = NULL,
  wwtp_column_mapping = NULL,
  wwtp_skip_rows = 0,
  wwtp_header_row = 1,
  wwtp_load_units = "kg",
  add_texas = FALSE,
  save_outputs = TRUE,
  cropland_threshold = NULL,
  verbose = TRUE
)
```

**Arguments**

scale	Character. Spatial scale: "county", "huc8", or "huc2"
year	Numeric. Year to analyze (available: start-2016 for NuGIS, 2016 for built-in WWTP)
nutrients	Character vector. Nutrients to analyze: c("nitrogen", "phosphorus") or subset
output_dir	Character. Output directory for results (default: "manureshed_results")
include_wwtp	Logical. Whether to include WWTP analysis (default: TRUE)
wwtp_year	Numeric. Year for WWTP data (default: same as year, only 2016 available built-in)
custom_wwtp_nitrogen	Character. Path to custom WWTP nitrogen file (for non-2016 years)
custom_wwtp_phosphorus	Character. Path to custom WWTP phosphorus file (for non-2016 years)
wwtp_column_mapping	Named list. Custom column mapping for WWTP data
wwtp_skip_rows	Numeric. Rows to skip in custom WWTP files (default: 0)
wwtp_header_row	Numeric. Header row in custom WWTP files (default: 1)
wwtp_load_units	Character. Units of WWTP loads: "kg", "lbs", "pounds", "tons" (default: "kg")
add_texas	Logical. Whether to add Texas HUC8 data (only for HUC8 scale, default: FALSE)
save_outputs	Logical. Whether to save results to files (default: TRUE)
cropland_threshold	Numeric. Custom cropland threshold for exclusion (optional)
verbose	Logical. Whether to print detailed progress messages (default: TRUE)

**Value**

List with all analysis results for specified nutrients

**Examples**

```
# Basic analysis using built-in data (2007-2016 WWTP available)
results_2016 <- run_builtin_analysis(
  scale = "huc8",
  year = 2016,
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = TRUE
)

# Analysis for earlier year (no WWTP available) - nitrogen only
results_2010 <- run_builtin_analysis(
  scale = "county",
  year = 2010,
```

```

    nutrients = "nitrogen",
    include_wwtp = FALSE
  )

  # Analysis for earlier year with WWTP now available
  results_2010 <- run_builtin_analysis(
    scale = "county",
    year = 2010,
    nutrients = "nitrogen",
    include_wwtp = TRUE # Now supported for 2010!
  )

  # Analysis for year before WWTP availability
  results_2005 <- run_builtin_analysis(
    scale = "huc8",
    year = 2005,
    nutrients = "phosphorus",
    include_wwtp = FALSE # No WWTP data before 2007
  )

```

---

run\_state\_analysis      *Run State-Level Analysis*

---

### Description

Run manureshed analysis for a specific state

### Usage

```

run_state_analysis(
  state,
  scale = "huc8",
  year = 2016,
  nutrients = c("nitrogen", "phosphorus"),
  include_wwtp = TRUE,
  output_dir = file.path(tempdir(), paste0("state_", tolower(state), "_results")),
  verbose = TRUE,
  ...
)

```

### Arguments

state	Character. Two-letter state abbreviation (e.g., "OH", "TX")
scale	Character. Spatial scale: "county", "huc8", or "huc2"
year	Numeric. Year to analyze
nutrients	Character vector. Nutrients to analyze
include_wwtp	Logical. Whether to include WWTP analysis

output_dir	Character. Output directory
verbose	Logical. Show progress messages
...	Additional arguments passed to run_built_in_analysis

**Value**

List with analysis results for the state

**Examples**

```
# Use Texas which has more data
texas_results <- run_state_analysis(
  state = "TX",
  scale = "county", # Use county for faster processing
  year = 2016,
  nutrients = "nitrogen", # Single nutrient for speed
  include_wwtp = TRUE
)

# California county-level analysis
ca_results <- run_state_analysis(
  state = "CA",
  scale = "county",
  year = 2010,
  nutrients = "nitrogen"
)
```

---

save\_analysis\_summary *Save Analysis Summary*

---

**Description**

Save comprehensive summary of analysis parameters and results

**Usage**

```
save_analysis_summary(results, file_path = NULL, format = "rds")
```

**Arguments**

results	List. Complete analysis results from workflow functions
file_path	Character. Output file path. If NULL, auto-generated
format	Character. Output format: "rds", "json", or "txt"

**Value**

Character. Path to saved file

## Examples

```
# Create analysis results first
results <- run_builtin_analysis(scale = "county", year = 2016)

# Save complete analysis summary
summary_path <- file.path(tempdir(), "analysis_summary_2016.json")
save_analysis_summary(results, summary_path, format = "json")
```

---

save\_centroid\_data      *Save Centroid Data*

---

## Description

Save centroid data to CSV file for transition probability analysis

## Usage

```
save_centroid_data(
  data,
  file_path = NULL,
  scale = "huc8",
  nutrient = "nitrogen",
  analysis_type = "centroids",
  year = format(Sys.Date(), "%Y")
)
```

## Arguments

data	Data frame. Data with centroid coordinates
file_path	Character. Output file path (should end in .csv). If NULL, auto-generated
scale	Character. Spatial scale for file naming
nutrient	Character. Nutrient type for file naming
analysis_type	Character. Analysis type for file naming
year	Numeric. Year for file naming

## Value

Character. Path to saved file

## Examples

```
# Create some example data first
results <- run_builtin_analysis(scale = "county", year = 2016, include_wwtp = TRUE)

# Save centroids for transition analysis
if ("integrated" %in% names(results) && "nitrogen" %in% names(results$integrated)) {
  centroids <- add_centroid_coordinates(results$integrated$nitrogen)
  save_centroid_data(centroids, scale = "county", nutrient = "nitrogen")
}
```

---

save\_plot

*Save Plot*

---

## Description

Save ggplot object to file with publication-quality settings

## Usage

```
save_plot(
  plot,
  file_path,
  width = 11,
  height = 6,
  dpi = 300,
  units = "in",
  device = NULL
)
```

## Arguments

plot	ggplot object. Plot to save
file_path	Character. Output file path
width	Numeric. Plot width in inches (default: 11)
height	Numeric. Plot height in inches (default: 6)
dpi	Numeric. Resolution in dots per inch (default: 300)
units	Character. Units for width and height (default: "in")
device	Character. Output device (auto-detected from file extension)

## Value

Character. Path to saved file

**Examples**

```
# Create a simple plot for demonstration
library(ggplot2)
p <- ggplot(mtcars, aes(x = mpg, y = hp)) + geom_point()

# Save with default settings (300 DPI, 11x6 inches)
save_plot(p, file.path(tempdir(), "test_plot.png"))

# Save with custom dimensions for presentation
save_plot(p, file.path(tempdir(), "presentation_plot.png"), width = 16, height = 9)

# Save as PDF for publication
save_plot(p, file.path(tempdir(), "publication_figure.pdf"), width = 8, height = 6)
```

---

save\_spatial\_data      *Save Spatial Data*

---

**Description**

Save spatial data to RDS file with standardized naming

**Usage**

```
save_spatial_data(
  data,
  file_path = NULL,
  scale = "huc8",
  nutrient = "both",
  analysis_type = "combined",
  year = format(Sys.Date(), "%Y")
)
```

**Arguments**

data	sf object. Spatial data to save
file_path	Character. Output file path (should end in .rds). If NULL, auto-generated
scale	Character. Spatial scale for file naming
nutrient	Character. Nutrient type for file naming ("nitrogen", "phosphorus", or "both")
analysis_type	Character. Analysis type for file naming
year	Numeric. Year for file naming

**Value**

Character. Path to saved file



## Examples

```
# Create some example results first
results <- run_builtin_analysis(scale = "county", year = 2016)

# Save with auto-generated filename
save_spatial_data(results$agricultural, scale = "county", year = 2016)

# Save with custom filename
save_spatial_data(results$agricultural,
  file.path(tempdir(), "my_results.rds"))
```

---

save\_transition\_matrix

*Save Transition Probability Matrix*

---

## Description

Save transition probability matrix to CSV with metadata

## Usage

```
save_transition_matrix(
  transition_df,
  file_path,
  nutrient,
  analysis_type = "combined",
  metadata = NULL
)
```

## Arguments

transition_df	Data frame. Transition probability matrix
file_path	Character. Output file path
nutrient	Character. Nutrient type
analysis_type	Character. Type of analysis
metadata	List. Additional metadata to include

## Value

Character. Path to saved file

**Examples**

```
# Create example analysis results first
results <- run_builtin_analysis(scale = "county", year = 2016, include_wwtp = TRUE)

# Save transition probabilities (only if integrated results exist)
if ("integrated" %in% names(results) && "nitrogen" %in% names(results$integrated)) {
  centroids <- add_centroid_coordinates(results$integrated$nitrogen)
  transitions <- calculate_transition_probabilities(centroids, "combined_N_class")
  save_transition_matrix(transitions,
                        file.path(tempdir(), "transitions_nitrogen.csv"),
                        "nitrogen")
}
```

---

summarize\_results      *Print Summary of Analysis Results*

---

**Description**

Print formatted summary of manureshed analysis results to the console. The summary includes analysis configuration parameters (scale, year, nutrients, WWTP inclusion), spatial coverage statistics, agricultural nutrient classifications with counts and percentages, WWTP integration metrics (if applicable), integrated classifications (if available), output file information, and processing time.

**Usage**

```
summarize_results(results, detailed = FALSE)
```

**Arguments**

results	List. Analysis results from <a href="#">run_builtin_analysis</a> or <a href="#">run_state_analysis</a> . Must contain at minimum: <ul style="list-style-type: none"> <li>• parameters: List with scale, year, nutrients, include_wwtp</li> <li>• agricultural: sf data frame with classification columns</li> </ul> Optional components: <ul style="list-style-type: none"> <li>• wwtp: WWTP analysis results</li> <li>• integrated: Integrated classification results</li> <li>• created_files or saved_files: Output file paths</li> </ul>
detailed	Logical. If TRUE, includes additional breakdown of integrated classifications showing combined agricultural-WWTP nutrient classes. If FALSE (default), shows only agricultural classifications and basic WWTP statistics.

## Details

The summary output is organized into sections:

**Analysis Configuration** Scale, year, nutrients analyzed, WWTP inclusion, state (if applicable)

**Spatial Coverage** Total number of spatial units analyzed

**Agricultural Classifications** Nitrogen and phosphorus classification counts and percentages

**WWTP Integration** Number of facilities and total loads by nutrient (if applicable)

**Integrated Classifications** Combined agricultural-WWTP classes (if detailed = TRUE)

**Output Files** Number and types of created files (if saved)

**Processing Time** Analysis duration in minutes (if available)

Classification names are cleaned for display (underscores replaced with spaces, line breaks removed). Percentages are rounded to one decimal place. All console output uses [message](#) and can be suppressed with [suppressMessages](#).

## Value

Invisibly returns the input `results` list unchanged. The function is called primarily for its side effect of printing a formatted summary to the console. The invisible return allows for piping operations while displaying the summary.

## See Also

[run\\_built\\_in\\_analysis](#) for generating analysis results, [quick\\_check](#) for quick validation, [compare\\_analyses](#) for comparing two result sets

## Examples

```
# Basic summary
results <- run_built_in_analysis(scale = "county", year = 2016)
summarize_results(results)

# Detailed summary with integrated classifications
results <- run_built_in_analysis(
  scale = "huc8",
  year = 2012,
  include_wwtp = TRUE
)
summarize_results(results, detailed = TRUE)

## Not run:
# This requires magrittr - never auto-run
library(magrittr)
results <- run_built_in_analysis(scale = "huc2", year = 2015) %>%
  summarize_results() %>%
  export_for_gis(output_dir = tempdir())

## End(Not run)
```

---

test\_osf\_connection     *Test OSF Connection*

---

**Description**

Test downloading a small dataset to verify OSF connectivity

**Usage**

```
test_osf_connection(verbose = TRUE)
```

**Arguments**

verbose                Logical. Show detailed test results

**Value**

Logical. TRUE if test successful

---

validate\_columns        *Validate Required Columns*

---

**Description**

Validate Required Columns

**Usage**

```
validate_columns(data, required_cols, data_type = "data")
```

**Arguments**

data                    Data frame to validate  
required\_cols          Character vector of required column names  
data\_type               Character description of data type for error messages

**Value**

Logical. TRUE if all columns present, stops with error otherwise

---

 wwtp\_aggregate\_by\_boundaries

*Aggregate WWTP Data by Spatial Boundaries*


---

**Description**

Aggregate WWTP loads by spatial units (counties, HUC8s, etc.)

**Usage**

```
wwtp_aggregate_by_boundaries(wwtp_sf, boundaries, nutrient, boundary_id_col)
```

**Arguments**

wwtp_sf	sf object. Spatial WWTP data
boundaries	sf object. Spatial boundaries for aggregation
nutrient	Character. "nitrogen" or "phosphorus"
boundary_id_col	Character. Name of boundary ID column

**Value**

Data frame with aggregated loads by spatial unit

---

 wwtp\_classify\_sources *Classify WWTP Sources by Load Size*


---

**Description**

Classify WWTP facilities into size categories based on annual nutrient loads

**Usage**

```
wwtp_classify_sources(wwtp_data, nutrient)
```

**Arguments**

wwtp_data	Data frame. WWTP data with load information
nutrient	Character. "nitrogen" or "phosphorus"

**Value**

Data frame with source\_class column added

**Examples**

```
# Load WWTP data first
wwtp_data <- load_builtin_wwtp("nitrogen", 2016)

# Classify nitrogen sources
classified_data <- wwtp_classify_sources(wwtp_data, "nitrogen")
table(classified_data$source_class)
```

---

wwtp_clean_data	<i>Clean WWTP Data</i>
-----------------	------------------------

---

**Description**

Clean and validate WWTP data for analysis  
Clean and validate WWTP data for analysis

**Usage**

```
wwtp_clean_data(wwtp_data, nutrient)

wwtp_clean_data(wwtp_data, nutrient)
```

**Arguments**

wwtp_data	Data frame. Raw or standardized WWTP data
nutrient	Character. "nitrogen" or "phosphorus"

**Value**

Data frame with cleaned WWTP data  
Data frame with cleaned WWTP data

**Examples**

```
## Not run:
# Clean user-loaded data will not run. They need to be supplied by users
clean_data <- wwtp_clean_data(raw_wwtp_data, "nitrogen")

# Clean OSF data (usually already clean, available 2007-2016)
osf_data <- load_builtin_wwtp("phosphorus", 2012)
clean_data <- wwtp_clean_data(osf_data, "phosphorus")

## End(Not run)

# Load and clean WWTP data
raw_wwtp_data <- load_builtin_wwtp("nitrogen", 2016)
clean_data <- wwtp_clean_data(raw_wwtp_data, "nitrogen")
```

---

`wwtp_filter_positive_loads`*Filter WWTP Data for Positive Loads*

---

**Description**

Filter WWTP data to include only facilities with positive nutrient loads

**Usage**

```
wwtp_filter_positive_loads(wwtp_data, nutrient)
```

**Arguments**

<code>wwtp_data</code>	Data frame. WWTP data
<code>nutrient</code>	Character. "nitrogen" or "phosphorus"

**Value**

Data frame with facilities having positive loads

---

`wwtp_process_complete` *Complete WWTP Processing Pipeline*

---

**Description**

Run complete WWTP processing pipeline for both nutrients

**Usage**

```
wwtp_process_complete(  
  nitrogen_path = NULL,  
  phosphorus_path = NULL,  
  column_mapping = NULL,  
  skip_rows = 0,  
  header_row = 1,  
  load_units = "kg",  
  verbose = TRUE  
)
```

**Arguments**

nitrogen\_path Character. Path to nitrogen WWTP data (if NULL, loads from OSF)  
 phosphorus\_path Character. Path to phosphorus WWTP data (if NULL, loads from OSF)  
 column\_mapping Named list. Custom column mapping for user data  
 skip\_rows Numeric. Rows to skip in user files  
 header\_row Numeric. Header row in user files  
 load\_units Character. Units of loads in user files  
 verbose Logical. Show processing messages

**Value**

List with processed nitrogen and phosphorus WWTP data

**Examples**

```
# Process built-in OSF data (2016 default)
wwtp_results_builtin <- wwtp_process_complete(
  nitrogen_path = NULL,      # Use built-in data
  phosphorus_path = NULL,   # Use built-in data
  verbose = TRUE
)

# Process custom user data
# wwtp_results_custom <- wwtp_process_complete(
#   nitrogen_path = "nitrogen_2020.csv",
#   phosphorus_path = "phosphorus_2020.csv",
#   load_units = "lbs"
# )

# Mixed: OSF for one nutrient, custom for another
# wwtp_results_mixed <- wwtp_process_complete(
#   nitrogen_path = NULL,      # Use OSF built-in
#   phosphorus_path = "custom_P.csv" # Use custom
# )
```

---

 wwtp\_to\_spatial

*Convert WWTP Data to Spatial Format*


---

**Description**

Convert WWTP data frame to sf spatial object

**Usage**

```
wwtp_to_spatial(wwtp_data, crs = 4326)
```



**Arguments**

`wwtp_data` Data frame. WWTP data with Lat/Long coordinates  
`crs` Numeric. Coordinate reference system (default: 4326 for WGS84)

**Value**

sf object with WWTP facilities as point geometries

**Examples**

```
# Load and convert to spatial format
wwtp_data <- load_built_in_wwtp("nitrogen", 2016)
wwtp_clean_data <- wwtp_clean_data(wwtp_data, "nitrogen")
wwtp_sf <- wwtp_to_spatial(wwtp_clean_data)

# Convert and transform to analysis CRS (without using pipe operator)
wwtp_sf <- wwtp_to_spatial(wwtp_clean_data)
wwtp_sf_transformed <- sf::st_transform(wwtp_sf, 5070) # Albers Equal Area Conic
```

# Index

## \* datasets

- CONUS\_STATES, 20
  - data\_sources, 23
  - KG\_TO\_TONS, 32
  - LBS\_TO\_KG, 32
  - LBS\_TO\_TONS, 32
  - MANURESHED\_CRIS, 42
  - P205\_TO\_P, 45
- add\_centroid\_coordinates, 4, 38
- add\_texas\_huc8, 4, 38
- agri\_classify\_complete, 5, 38
- agri\_classify\_complete\_custom, 6
- agri\_classify\_nitrogen, 7, 38
- agri\_classify\_nitrogen\_custom, 7
- agri\_classify\_phosphorus, 9, 38
- agri\_classify\_phosphorus\_custom, 9
- agri\_process\_nugis, 10, 38
- batch\_analysis\_enhanced, 11
- batch\_analysis\_parallel, 12
- batch\_analysis\_years, 13, 38
- benchmark\_analysis, 14
- calculate\_cropland\_threshold, 15
- calculate\_transition\_probabilities, 16, 38
- check\_built\_in\_data, 16, 17, 38
- citation\_info, 17
- clean\_category\_names, 18, 39
- clean\_text, 18
- clear\_data\_cache, 19, 38
- compare\_analyses, 19, 59
- CONUS\_STATES, 20
- convert\_load\_units, 20, 38
- create\_analysis\_report, 21
- create\_classification\_summary, 22, 39
- create\_network\_plot, 22, 38
- data\_sources, 23
- download\_all\_data, 24, 38
- download\_osf\_data, 24, 38
- export\_for\_gis, 25
- export\_for\_policy, 26
- export\_for\_publication, 26
- filter\_by\_state, 27
- format\_huc8, 27, 39
- get\_cropland\_threshold, 28, 39
- get\_nutrient\_colors, 28, 39
- get\_state\_boundaries, 29, 38
- get\_state\_fips, 29
- health\_check, 17, 30
- integrate\_complete, 30, 38
- integrate\_wwtp\_agricultural, 31, 38
- KG\_TO\_TONS, 32
- LBS\_TO\_KG, 32
- LBS\_TO\_TONS, 32
- list\_available\_years, 33
- load\_built\_in\_boundaries, 33, 38
- load\_built\_in\_nugis, 34, 38
- load\_built\_in\_wwtp, 34, 38
- load\_user\_wwtp, 36, 38
- manureshed, 37
- manureshed-package (manureshed), 37
- MANURESHED\_CRIS, 42
- map\_agricultural\_classification, 38, 43
- map\_wwtp\_columns, 43
- map\_wwtp\_influence, 38, 44
- map\_wwtp\_points, 38, 44
- message, 59
- P205\_TO\_P, 45
- plot\_absolute\_changes, 39, 45

plot\_before\_after\_comparison, [39](#), [46](#)  
plot\_impact\_ratios, [39](#), [46](#)

quick\_analysis, [38](#), [47](#)  
quick\_check, [48](#), [59](#)  
quick\_state\_analysis, [49](#)

run\_built\_in\_analysis, [38](#), [50](#), [58](#), [59](#)  
run\_state\_analysis, [52](#), [58](#)

save\_analysis\_summary, [53](#)  
save\_centroid\_data, [54](#)  
save\_plot, [55](#)  
save\_spatial\_data, [56](#)  
save\_transition\_matrix, [57](#)  
summarize\_results, [58](#)  
suppressMessages, [59](#)

test\_osf\_connection, [60](#)

validate\_columns, [60](#)

wtp\_aggregate\_by\_boundaries, [61](#)  
wtp\_classify\_sources, [38](#), [61](#)  
wtp\_clean\_data, [38](#), [62](#)  
wtp\_filter\_positive\_loads, [63](#)  
wtp\_process\_complete, [63](#)  
wtp\_to\_spatial, [64](#)