A practitioner's guide to geospatial analytics: the no-code approach

Advancements in 5G infrastructure, AI-supported analytics tools, and machine learning are fueling the geospatial analytics market. **In fact, it's expected to grow 12.5% for the forecasted period of 2022-2032**. As more and more devices and applications are able to log location information, the value of geospatial analysis becomes more widespread and the use cases multiply.

In business, geospatial analytics is becoming a major gateway to deeper insights — helping teams and their leaders make better decisions for site selection, supply chain optimization, environment management, fraud detection, and more. Historically, accessing and analyzing this invaluable data has required niche expertise, demanding deep experience with a wide array of data sources and libraries, as well as coding skills.

Harvard's Center for Geographic Analysis (CGA) and KNIME have teamed up to enable non-expert users to unlock the potential of geospatial data by accessing, blending, and analyzing it through a no-code environment. This guide covers an overview of geospatial basics as well as how to best leverage KNIME's geospatial analysis extension. As a companion to this guide, you'll also find reusable and reproducible data and blueprints you can use and adapt to your needs.

About Harvard University's Center for Geographic Analysis

The Center for Geographic Analysis (CGA), which works with entities across Harvard, strengthens university-wide geographic information systems (GIS) infrastructure and services. It provides a common platform for the integration of spatial data from diverse sources and knowledge from multiple disciplines and enables scholarly research that uses, improves, or studies geospatial analysis techniques. In addition, the CGA improves the teaching of GIS and geospatial data science at all levels across the University.

Geospatial basics

Let's start with basic concepts of geospatial analytics, such as different formats and types.

Data formats

To get started, you need to get your hands on geospatial data. One way to do this is by reading data in a special geometric format. In the following section, we introduce the most common ones that are also supported in KNIME.

Latitude and longitude

If you have ever used Google Maps or another navigation system, you should have come across **latitude and longitude**. Given a fixed reference line from the center of the earth through the point of interest, latitude describes the degree of the angle between the line and the equatorial plane, whereas longitude describes the degree of the angle between the reference line and a line from the Earth center through Greenwich — which acts as reference point. In other words, latitude describes the North to South position and longitude describes the East to West. The combination of these two coordinates can be used to specify any position on Earth.

Well-known text

The most commonly used human-readable format to represent vector geometry objects is the well-known text format (**WKT**). The format was specified by the **Open Geospatial Consortium** and specified in their **Simple Feature Access** standard. The binary equivalent of WKT is the well-known binary format (**WKB**), which is a more compact but not human-readable representation of the WKT.

GeoJSON

GeoJSON is a **JSON**-based format that represents vector geometry objects along with other non-geometric features, such as strings and numbers. The benefit of GeoJSON is that it can not only store geometric information, but also non-geometric attributes such as a country name and population size.

Shapefile

The **shapefile** format, developed by Esri, is commonly used by different geographic information systems. Like GeoJSON, shapefiles can store not only geometric information but also additional non-geometric attributes (or metadata). However, the name might be misleading because a shapefile does not consist of a single file but of several files. Of these files, the shape format (.shp), shape index format (.shx), and attribute format (.dbf) files are mandatory. Usually, when shapefiles are provided, they are zipped. But you don't need to extract these files to read them in KNIME Analytics Platform since the **GeoFile Reader** node supports reading of zipped shapefiles.

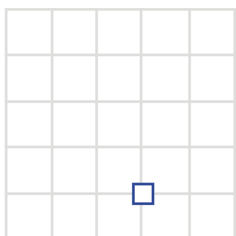
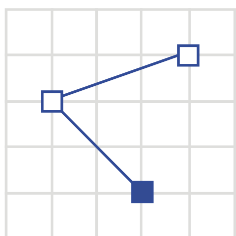
GeoPackage

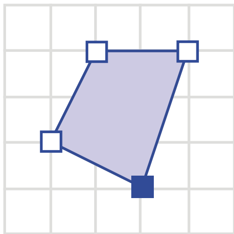
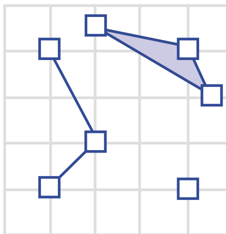
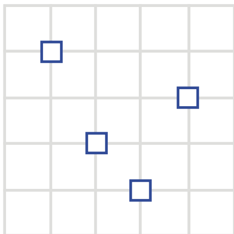
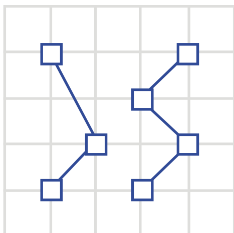
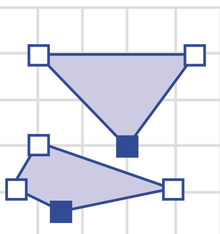
The **GeoPackage** format stores a SQLite database to store multi-layered vector features and their attributes. It is standardized by the **Open Geospatial Consortium**.

Data types

In general, geospatial vector data uses different object types to represent different shapes, such as streets, borders, or points of interest. These types can either represent a single object such as a street, or a collection of multiple objects such as a university campus.

The following table lists the most commonly used objects and their WKT type.

WKT Type	Image	Example
Point		POINT (30 10)
LineString		LINESTRING (30 10, 10 30, 40 40)

Polygon		POLYGON ((30 10, 40 40, 20 40, 10 20, 30 10))
GeometryCollection		GEOMETRYCOLLECTION (POINT (40 10), LINESTRING (10 10, 20 20, 10 40), POLYGON ((40 40, 20 45, 45 30, 40 40)))
MultiPoint		MULTIPOINT ((10 40), (40 30), (20 20), (30 10))
MultiLineString		MULTILINESTRING ((10 10, 20 20, 10 40), (40 40, 30 30, 40 20, 30 10))
MultiPolygon		MULTIPOLYGON (((30 20, 45 40, 10 40, 30 20)), ((15 5, 40 10, 10 20, 5 10, 15 5)))

Images and examples copied from Wikipedia

Coordinate reference system

A **coordinate reference system (CRS)** defines a location on Earth. The two major types of coordinate reference systems are geographic and projected. A **geographic coordinate system** models the Earth as a three-dimensional sphere and uses angles to describe locations. One of the most commonly used geographic coordinate systems is the **WGS 84** (**EPSG:4326**). It is not only the default for GPS but also used in various databases (e.g., Snowflake and PostGIS), file formats (e.g., GeoJSON) and most web mapping libraries. It is also the default coordinate reference system in KNIME used whenever you do not specify a dedicated coordinate reference system.

A **projected coordinate system** on the other hand models the earth as a two-dimensional plane with locations identified by x and y coordinates. One of the most commonly used projected coordinate systems is the **Web Mercator projection** (**EPSG:3857**). It's the default in most web mapping libraries, including **Google Maps** and **OpenStreetMap**.

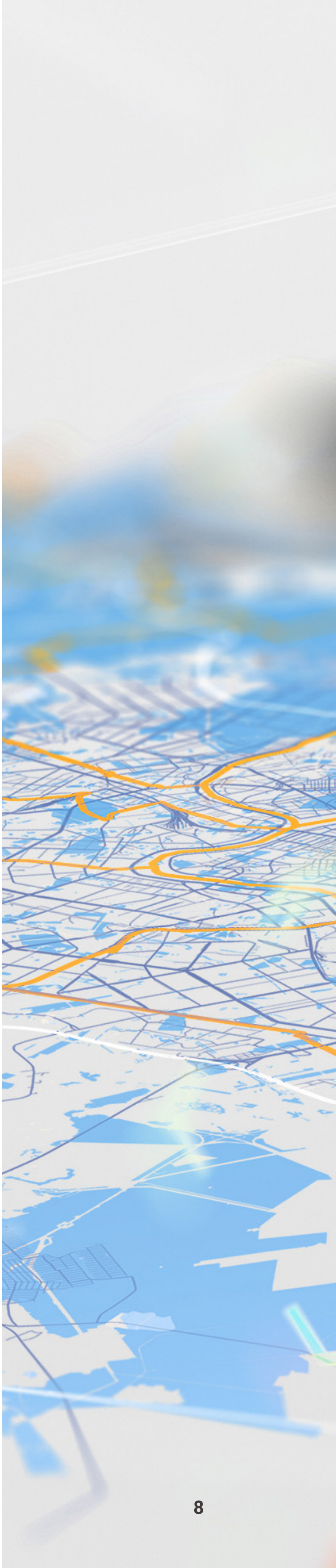
The challenge with geospatial analysis is that the Earth is a sphere, but we usually look at it on a flat map. This means the three-dimensional Earth has to be projected onto a two-dimensional coordinate system, which we discuss in the next section.

Projection

Projection is the transformation of the three-dimensional Earth onto a two-dimensional map. The transformation poses a problem since it will always result in some distortions (explained in **this video**). Among others, these distortions can affect the following properties:

- Area
- Form
- Distance
- Direction

As an example of these distortions, you can see different images of Germany using different projections that preserve different properties. The different images have been created in this **KNIME workflow**.



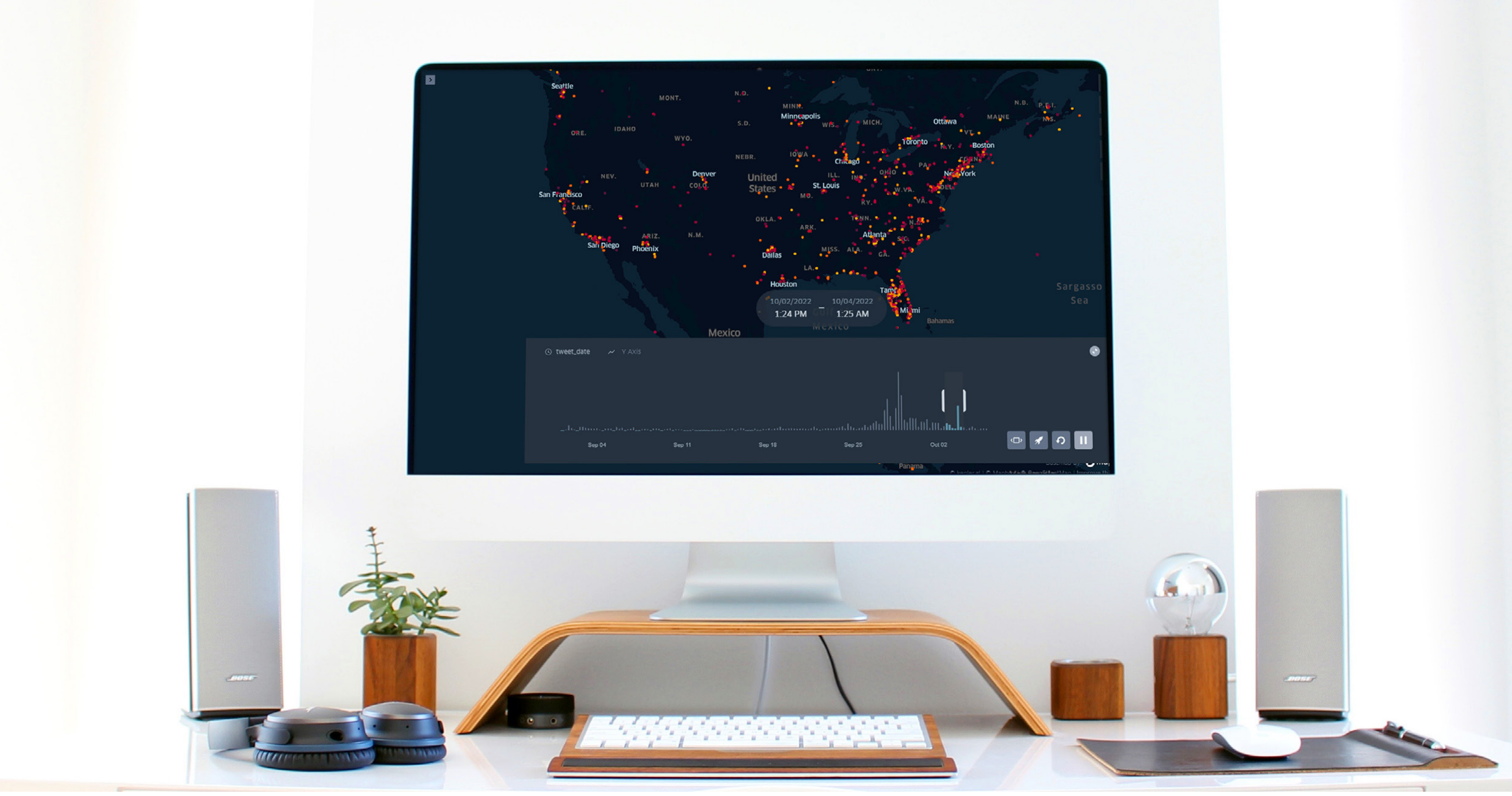
Mercator projection	Azimuthal equidistant	Equal earth projection
		
This shape should look familiar to you since many online mapping services such as Google and OpenStreetMap use a variant of the Mercator projection.	With the **Azimuthal equidistant projection**, Germany looks a bit wider than with the Mercator projection.	The **equal earth projection** looks much wider than the Mercator projection.
Pros: Preserves the direction (e.g., North is always up.). **Cons:** Distorts the shape and size with increasing distance from the equator.	**Pros:** Preserves the distance between two points on the map. **Cons:** Distortions will increase with the distance between the plotted objects.	**Pros:** Preserves the area and provides a visually pleasing appearance similar to those found on a globe. **Cons:** Tries to preserve all aspects, but in doing so is not perfect in any.

Please be aware that different projections use different units (e.g., meters or degrees).

Each projection comes with a trade off between shape, distance, direction, and area:

- **Conformal** projections preserve the **shape** of features. They are good for general-purpose reference mapping where a place should look familiar.
- **Equal-area** projections preserve the area of a feature. You should use this type of projection whenever you want to map density information (e.g., number of persons per square kilometer).
- **Equidistant** projections preserve the distance between two features. Whenever you want to show how far things are from each other you should use this kind of projection.
- **Compromise** projections (as the name suggests) try to find a compromise by preserving different properties to a certain extent.

In the end, choosing the right projection depends on the goal of your analysis.



Geospatial analytics in KNIME

This section will get you started with applying geospatial data in KNIME Analytics Platform. It is also intended to give you a short overview of the various node categories and to provide tips for working with nodes.

Installation

To get started, you need to install the geospatial extension (detailed in this **blog post**). The blog post also covers how to build an interactive geospatial application without code.

Getting the data

The first step in the geospatial analysis journey is to get the information into a KNIME workflow, which is described in the following section.

Geospatial files

KNIME provides various reader nodes to read all of the **previously mentioned file formats**. The **GeoFile Reader** node for example supports reading of **Shapefile (.shp)**, zipped Shapefiles(.zip) with a single Shapefile, **GeoJSON (.geojson)**, or single-layer **Geopackage (.gpkg)** files. Reading of multi layer Geopackage files can be done with the **GeoPackage Reader** node.

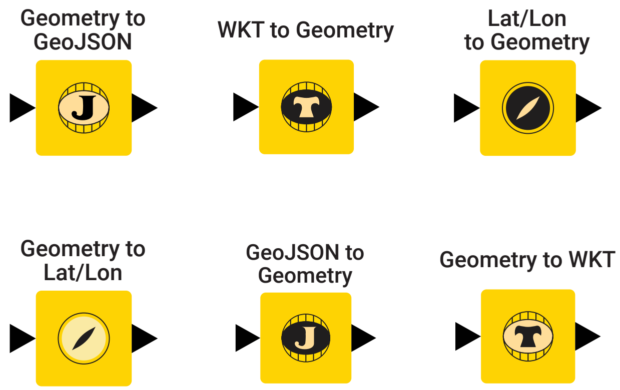
All reader nodes support local file paths and web URLs. For example, to load a GeoJSON file from **geojson.xyz**, you would enter: **http://d2ad6b4ur7yvpq.cloudfront.net/naturalearth-3.3.0/ne_110m_land.geojson**.

For an example on how to read different file formats from within KNIME Analytics Platform, go to this example **workflow**.

The writer nodes on the other hand allow you to write out the result of your analysis in a geospatial format that is supported by most geospatial tools. The **GeoPackage Writer** allows you to append new layers to an already existing file. To do so, simply provide the path of the file and the new layer name.

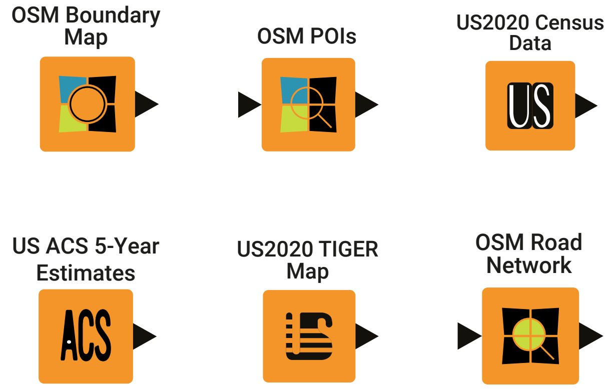
Other files with geospatial data

If you have a non-geospatial data file such as CSV or Parquet that contains geospatial information (like longitude/latitude, a WKT, or GeoJSON string) you can use one of the Spatial Conversion nodes.



Open dataset

Another way to get geospatial data into KNIME is by using the open dataset nodes. These reach out to external services such as **OpenStreetMap** or the **US Census Bureau** to get **points of interest**, **boundaries** (for countries or states), and **data about communities** in America.



When working geospatial data from different data sources, make sure to harmonize the CRS using the **Projection** node.

Geospatial data type

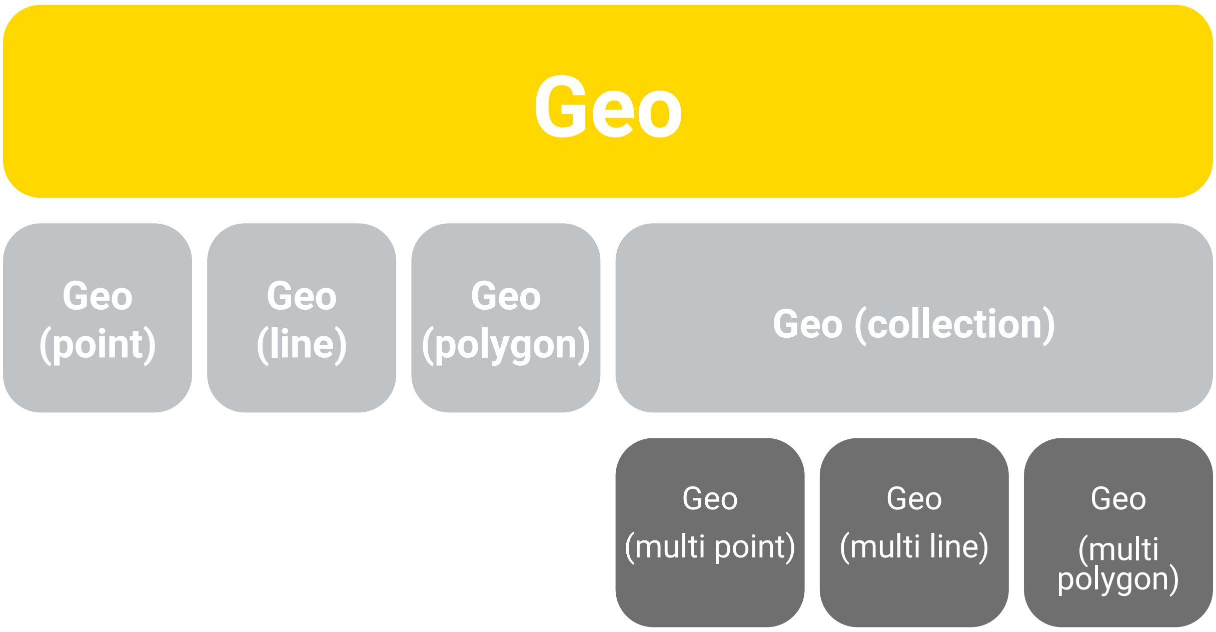
Once you have read in geospatial data, it is represented in a KNIME table using the new geospatial data types. KNIME supports all of the data types, which are also supported by WKT (For details, see the table in the '**Data types' section**). The following table shows the mapping between the KNIME and the corresponding WKT type.

KNIME Type	WKT Type
Geo (point)	**Point**
Geo (line)	**LineString**
Geo (polygon)	**Polygon**
Geo (collection)	**GeometryCollection**
Geo (multi point)	**MultiPoint**
Geo (multi line)	**MultiLineString**
Geo (multi polygon)	**MultiPolygon**

Type hierarchy

The geospatial data types in KNIME follow a hierarchy in accordance with the **OGC Geometry class hierarchy**. The most generic geospatial type is the Geo type. It can be used to represent all geometric types. The Geo type splits up into more specific single value types: Geo (point), Geo (line), and Geo (polygon), as well as the generic collection type Geo (collection). The Geo (collection) type splits up into more specific collection types: Geo (multi point), Geo (multi line), and Geo (multi polygon).

A KNIME node, such as the Projection node that supports columns of type Geo as input, supports all geospatial data types whereas a node that expects a Geo (point) such as the reverse geocoding node only supports columns of type Geo (point).



In general, you can see the type of column for a **KNIME data table**, as described **here**. You can also find a workflow with all supported data types and their type hierarchy **here**.

Processing

Once you have the geospatial data in a KNIME table, you might want to bring it into the right format (e.g., by harmonizing the coordinate reference system or by representing complex polygons with their representative point). In addition, you might want to extract information from the geospatial objects, such as the area of a country, length of a street, or distance between two objects.

Transformation

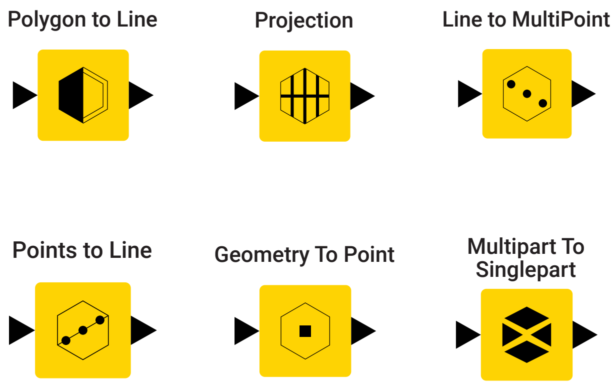
The transformation nodes in KNIME allow you to transform a geospatial object into another data type. One of the most common operations is the transformation of a Geometry (such as a polygon into a point by using the **Geometry to Point** node) which either computes the centroid or representative point of a given geometry. The major difference is that the representative point is guaranteed to be within the given geometry. Another very important transformation is the projection, which is covered in the next section.

Projection

In KNIME, the default **CRS** is **EPSG:4326**. In order to change the projection, you can use the **Projection** node. The node allows you to enter the CRS by entering an **EPSG code** or a **CRS WKT string**. For a selection of projections that preserve different properties, see this **Wikipedia article**. Once you have found the appropriate projection name or coordinate reference system, you can search for its EPSG code at **https://epsg.io/**. To do so, simply type the projection name into the search field (e.g., **Pseudo-Mercator**). The result page will show you the

EPSG code that you can enter in this field (EPSG:3857) and also the distance unit (e.g., meter or degree and the area of use where the projection works best).

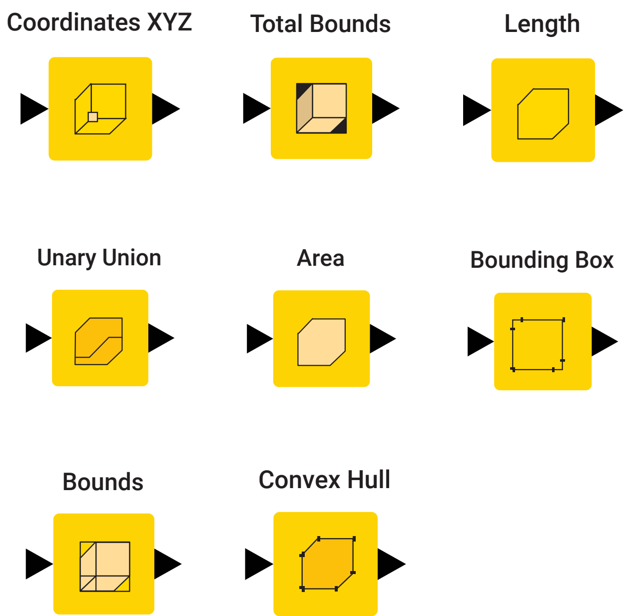
If you are looking for a projection for a specific area, try out the **Projection wizard page**, which suggests projection with specific properties for a defined area on the globe. Once you have found the appropriate projection, you can view the CRS WKT string by clicking on the **WKT** link next to the suggested projection name.



Calculation

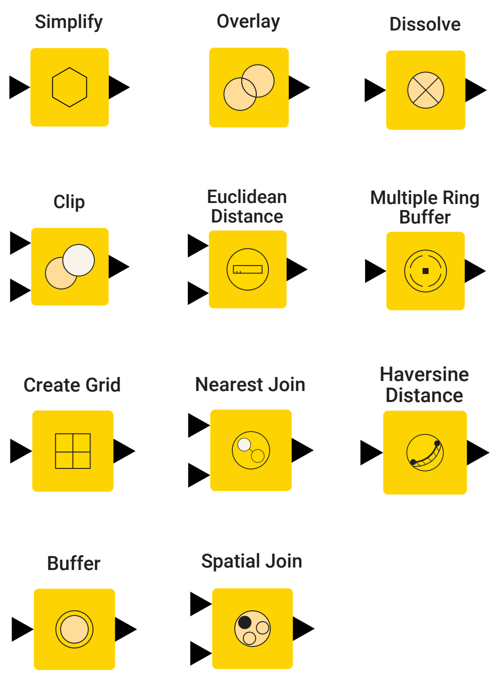
KNIME provides many different nodes that allow you to perform calculations, such as computing the **bounding box** or the **convex hull** of a geospatial object or to extract its **coordinates** or **total bounds**.

When computing the area or length of a geospatial object using the **Area** or **Length** node, you should consider using an **equal-area projection** such as the Equal Earth projection (**EPSG:8857**). The CRS of the input column defines the unit of the result column of these nodes and might be square meter or square miles.



Manipulation

The manipulation nodes allow you to perform various operations, such as computing a (multi) buffer around a given geospatial object or joining them based on their distance or relationship to each other using the **Nearest Join** or **Spatial Join**. If you want to perform geometric set operations (e.g., cutting out parts from a given geospatial object) you can use the **Overlay** node. If you are working with complex polygons, you might want to consider using the **Simplify** node to remove the number of points in the polygon and thus, speeding up its processing.

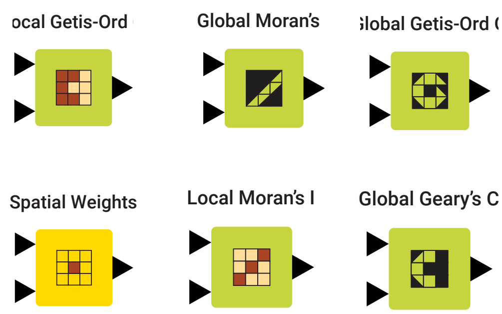


Analysis

The geospatial extension provides many different nodes for exploratory spatial data, location analysis, and spatial modeling.

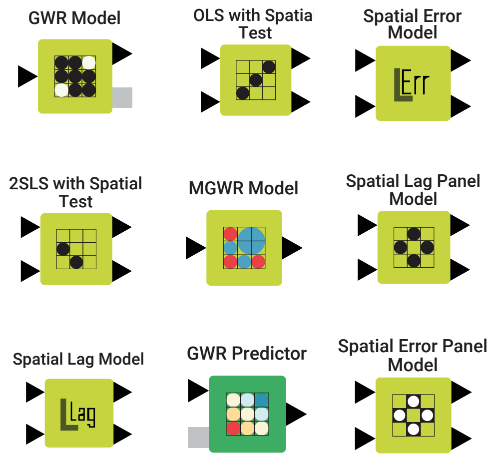
Exploratory analysis

This group of nodes can be used to compute different **spatial auto-correlations**, which can be used to identify if closely located geospatial objects are correlated or not. In addition, you can also perform clustering analysis (e.g., finding hot spots in your data).



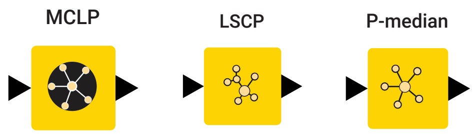
Modeling

To get more information about spatial relationships between geospatial objects, you can use the various spatial modeling nodes.



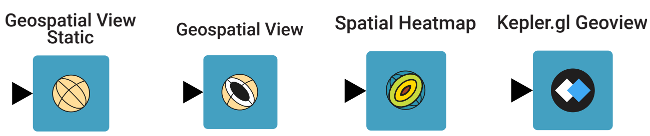
Location analysis

Location analysis is about discovering optimal site locations (e.g., optimal places for warehouses or hospitals to distribute goods or services).



Visualization

When working with geospatial data, it is important to view it on a map. KNIME provides various visualization nodes for different use cases:



The **Geospatial View Static** node provides a static view of the provided geospatial objects. It is also the only node that draws the objects in the input CRS, and is thus a good choice for **Choropleth maps** (as demonstrated in this **example workflow**).

The **Geospatial View** node provides an interactive map that lets you zoom in and out and select different objects. The geospatial objects are drawn on top of different base maps which is why the objects are projected to the base maps CRS prior visualization. The node provides various options, such as coloring and sizing, which are preserved in the node configuration.

The **Spatial Heatmap** node is based on the Geospatial View node but has an additional setting that allows you to specify a weight column which defines the "heat" of a data point.

The **Kepler.gl Geoview** node allows you to plot large amounts of data and is best suited to visualize time series data (as shown in this **example workflow**).

> The geometric elements are drawn in the order they appear in the input table. For example, if you want to show points within a polygon, you'll want to have the points drawn last on top of the polygon.

Applying the data to deliver deeper insights

As you can see, there's no need to be an expert in geospatial analysis in order to start working with the data at hand. With geospatial-specific nodes, access to predefined workflows, and drag-and-drop visual programming capabilities, KNIME equips both the non-expert and seasoned geographic information systems professional — so that any team or business leader looking to reap the benefits of geospatial analysis can do so.

And now that you know which KNIME nodes are available to help you analyze geospatial data and how, it's time to take advantage and start delivering new, deeper insights to your organization. For a hands-on tutorial that dives deeper into KNIME Analytics Platform and provides steps to visualize your geospatial data with workflows, visit **this blog**. Or, you can learn more about KNIME's geospatial analytics capabilities here:

- **Geospatial Analytics Extension for KNIME**
- **Geospatial Analytics Examples on the KNIME Hub**

Additionally, you can **contact us** for more information on how KNIME can accelerate geospatial analysis within your organization.

About KNIME

KNIME AG is an international company with offices in Zurich, Berlin, Konstanz, Austin, and online. Its analytics platform is an open-source software with an intuitive, visual interface that allows users to build analyses of any complexity level — from automating spreadsheets, to ELT, to predictive modeling and machine learning.

KNIME's enterprise software, KNIME Business Hub, provides organizations with a single, scalable environment to securely collaborate and share best practices, as well as deploy and monitor their analytical workflows. The scalable, cloud-native architecture and team-controlled execution enables fast community adoption and reduces the burden on central IT.

This suite of features enables organizations to build vibrant data science communities and accelerate the spread of data-driven decisioning.