DEMONSTRATION 1: Analyzing movement data with a (raster) environmental layer

Exercise based on data stored in the EURODEER database.
Land cover data source: Corine Land Cover project (http://land.copernicus.eu/pan-european/corine-land-cover)

Introduction

The advancement in movement ecology from a data perspective can reach its full potential only by combining the technology of animal tracking with the technology of other environmental sensing programmes. Ecology is fundamentally spatial, and animal ecology is obviously no exception. Any scientific question in animal ecology cannot overlook the dynamic interaction between individual animals or populations, and the environment in which the ecological processes occur. Movement provides the mechanistic link to explain this complex ecosystem interaction, as the movement path is dynamically determined by external factors, through their effect on the individual's state and the life-history characteristics of an animal. Therefore, most modelling approaches for animal movement include environmental factors as explanatory variables.
In these examples we will explore some simple analysis performed with spatial SQL into the database that relate animal movements based on GPS tracking data with land cover/use data.

Create database views for spatial representations of animal movement

Generate a view for the convex hull of all animals

```
CREATE OR REPLACE VIEW demo_florida.view_convexhull AS
SELECT
  animals_id,
  st_convexhull(st_collect(geom))::geometry(Polygon,4326) AS geom
 FROM
   demo_florida.gps_data_animals
 WHERE
   gps_validity_code = 1
 GROUP BY
   animals_id;
```

Generate a view for the trajectories of all animals

```
CREATE OR REPLACE VIEW demo_florida.view_trajectories AS
SELECT
  animals_id,
  st_makeline(foo.geom)::geometry(LineString,4326) AS geom
FROM (
  SELECT
    animals_id,
    geom
  FROM
    demo_florida.gps_data_animals
  WHERE
    gps_validity_code = 1
  ORDER BY
    animals_id,
    acquisition_time) foo
GROUP BY
  foo.animals_id;
```

Generate a view for the monthly convex hull of animal 782

```
CREATE OR REPLACE VIEW demo_florida.view_convexhull_monthly AS
SELECT
  row_number() over() AS id,
  animals_id,
  extract(month FROM acquisition_time) AS months,
  st_convexhull(st_collect(geom))::geometry(Polygon,4326) AS geom
 FROM
  demo_florida.gps_data_animals
 WHERE
  gps_validity_code = 1 AND
  animals_id = 782
 GROUP BY
  animals_id,
   extract(month FROM acquisition_time);
```

Set up raster layer into the database

Import land cover layer

> raster2pgsql.exe -C -t 128x128 -M -r C:/land_cover/corine_land_cover_2006.tif demo_florida.ndvi_modis | psql.exe -d eurodeer_db -U postgres -p 5432

Meaning of raster2pgsql parameters

- -C: new table
- -t: divide the images in tiles
- -M: vacuum analyze the raster table
- -r: Set the constraints for regular blocking

Create a table from an existing (larger) DB layer - LAND COVER

```
CREATE TABLE demo_florida.land_cover (rid SERIAL primary key, rast raster);

CREATE INDEX land_cover_rast_idx
  ON demo_florida.land_cover
  USING GIST (ST_ConvexHull(rast));

INSERT INTO demo_florida.land_cover (rast)
SELECT
  rast
FROM
  env_data.corine_land_cover_2006,
  main.study_areas
WHERE
  st_intersects(rast, ST_Expand(st_transform(geom, 3035), 5000)) AND
  study_areas_id = 1;

SELECT AddRasterConstraints('demo_florida'::name, 'land_cover'::NAME, 'rast'::name);
```

Export the layer to tiff

Create a new table with all reaster unioned, add constraints, export to TIFF with GDAL, drop the table CREATE TABLE demo_florida.land_cover_export(rast raster);

```
INSERT INTO
  demo_florida.land_cover_export
SELECT
  st_union(rast) AS rast
FROM
  demo_florida.land_cover;

SELECT AddRasterConstraints('demo_florida'::name, 'land_cover_export'::name, 'rast'::name);
```

Export with GDAL_translate

> gdal_translate -of GTIFF "PG:host=eurodeer2.fmach.it dbname=eurodeer_db user='postgres' schema=demo_florida table=land_cover_export mode=2" C:\Users\User\Desktop\Florida\land_cover.tif

Remove the unioned table

```
DROP TABLE demo_florida.land_cover_export;
```

Data analysis

Intersect the fixes with the land cover layer for the animal 782

```
SELECT
  st_value(rast,st_transform(geom, 3035)) as lc_id
FROM
  demo_florida.gps_data_animals,
  demo_florida.land_cover
WHERE
  animals_id = 782 AND
  gps_validity_code = 1 AND
  st_intersects(st_transform(geom, 3035), rast);
```

Calculate the percentage of each land cover class for fixes of the animal 782

```
WITH locations_landcover AS
    (
    SELECT
      st_value(rast,st_transform(geom, 3035)) AS lc_id
    FROM
      demo_florida.gps_data_animals,
      demo_florida.land_cover
     WHERE
      animals_id = 782 AND
      gps_validity_code = 1 AND
      st_intersects(st_transform(geom, 3035), rast)
    )
SELECT
  lc_id,
  label3,
  (count(*) * 1.0 / (SELECT count(*) FROM locations_landcover))::numeric(5,4) AS percentage
FROM
```

```
  locations_landcover,
  env_data.corine_land_cover_legend
WHERE
  grid_code = lc_id
GROUP BY
  lc_id,
  label3
ORDER BY
  percentage;
```

Intersect the convex hull of animal 782 with the land cover layer

```
SELECT
  (stats).value AS grid_code,
  (stats).count AS num_pixels
FROM
  (
  SELECT
    ST_valuecount(ST_union(st_clip(rast ,st_transform(geom,3035)))) AS stats
  FROM
    demo_florida.view_convexhull,
    demo_florida.land_cover
  WHERE
    animals_id = 782 AND
    st_intersects (rast, st_transform(geom,3035))
  ) a
```

Calculate the percentage of each land cover class in the convex hull for the animal 782

```
WITH convexhull_landcover AS
    (
    SELECT
      (stats).value AS lc_id,
      (stats).count AS num_pixels
    FROM
      (
      SELECT
        ST_valuecount(ST_union(st_clip(rast ,st_transform(geom,3035))))  stats
      FROM
        demo_florida.view_convexhull,
        demo_florida.land_cover
      WHERE
        animals_id = 782 AND
        st_intersects (rast, st_transform(geom,3035))
      ) AS a
    )
SELECT
  lc_id,
  label3,
  (num_pixels * 1.0 / (sum(num_pixels)over()))::numeric(5,4) AS percentage
FROM
  convexhull_landcover,
  env_data.corine_land_cover_legend
WHERE
  grid_code = lc_id
ORDER BY
  percentage DESC;
```

Intersect the fixes for males vs female with the land cover layer

```
SELECT
  sex,
  st_value(rast,st_transform(geom, 3035)) AS lc_id,
  count(*) AS number_locations
FROM
  demo_florida.gps_data_animals,
  demo_florida.land_cover,
  main.animals,
  env_data.corine_land_cover_legend
WHERE
  animals.animals_id = gps_data_animals.animals_id AND
  gps_validity_code = 1 AND
  st_intersects(st_transform(geom, 3035), rast)
GROUP BY
  sex, lc_id
ORDER BY
  lc_id;
```

Calculate the percentage of different land cover classes for all the monthly convex hulls of the animal 782

```
WITH convexhull_landcover AS
    (
    SELECT
      months,
```

```
      (stats).value AS lc_id,
      (stats).count AS num_pixels
    FROM (
      SELECT
        months,
        ST_valuecount(ST_union(st_clip(rast ,st_transform(geom,3035))))  stats
      FROM
        demo_florida.view_convexhull_monthly,
        demo_florida.land_cover
      WHERE
        st_intersects (rast, st_transform(geom,3035))
      GROUP BY
        months) a
    )
SELECT
  months,
  label3,
  (num_pixels * 1.0 / (sum(num_pixels) over (PARTITION BY months)))::numeric(5,4) AS percentage
FROM
  convexhull_landcover,
  env_data.corine_land_cover_legend
WHERE
  grid_code = lc_id
ORDER BY
  label3, months;
```

Calculate the percentage of each land cover class for all the fixes of animal 782

```
WITH locations_landcover AS
    (
    SELECT
      sex,
      st_value(rast,st_transform(geom, 3035)) AS lc_id,
      count(*) AS number_locations
    FROM
      demo_florida.gps_data_animals,
      demo_florida.land_cover,
      main.animals
     WHERE
      animals.animals_id = gps_data_animals.animals_id AND
      gps_validity_code = 1 AND
      st_intersects(st_transform(geom, 3035), rast)
    GROUP BY sex, lc_id
    )
SELECT
  sex,
  label3,
  (number_locations *1.0 / sum(number_locations) OVER (partition by sex))::numeric(5,4) AS percentage
FROM
  locations_landcover,
  env_data.corine_land_cover_legend
WHERE
  grid_code = lc_id
ORDER BY
  sex, percentage DESC;
```

DEMONSTRATION 2: Analyzing location data with a time series of environmental layers

Exercise based on data stored in the EURODEER database.
NDVI data source: MODIS NDVI (http://modis-land.gsfc.nasa.gov/vi.html), in a version (smoothed, weekly) downloaded from Boku University Portal](http://ivfl-info.boku.ac.at/index.php/eo-data-processing

Introduction

Animal locations are not only spatial, but are fully defined by spatial and temporal coordinates (as given by the acquisition time). Logically, the same temporal definition also applies to environmental layers. Some characteristics of the landscape, such as land cover or road networks, can be considered static over a large period of time and these environmental layers are commonly intersected with animal locations to infer habitat use and selection by animals. However, many characteristics actually relevant to wildlife, such as vegetation biomass or road traffic, are indeed subject to temporal variability (on the order of hours to weeks) in the landscape, and would be better represented by dynamic layers that correspond closely to the conditions actually encountered by an animal moving across the landscape. Nowadays, satellite-based remote sensing can provide high temporal resolution global coverage of medium-resolution images that can be used to compute a large number of environmental parameters very useful to wildlife studies. One of the most common set of environmental data time series is the Normalized Difference Vegetation Index (NDVI), but other examples include data sets on snow, ocean primary productivity, surface temperature, or salinity. Snowcover, NDVI, and sea surface temperature are some examples of indexes that can be used as explanatory variables in statistical models or to parametrize bayesian inferences or mechanistic models. The main shortcoming of such remote-sensing layers is the relatively low spatial resolution, which does not fit the current average bias of wildlife-tracking GPS locations (less than 20 m), thus potentially leading to a spatial mismatch between the *animal-based information and the environmental layers (note that the resolution can still be perfectly fine, depending on the overall spatial variability and the species and biological process under study). Higher-resolution images and newtypes of information (e.g. forest structure) are presently provided by newtypes of sensors, such as those from lidar, radar, or hyper-spectral remote-sensing technology and the newSentinel 2 (optical data). The newgeneration of satellites will probably require dedicated storage and analysis tools (e.g. Goggle Earth Engine) that can be related to the Big Data framework. Here, we will explore some simple example of spatio-temporal analyses that involve the interaction between GPS data and NDVI time series.*

The MODIS (Moderate Resolution Imaging Spectroradiometer) instrument operates on the NASA's Terra and Aqua spacecraft. The instrument views the entire earth surface every 1 to 2 days, captures data in 36 spectral bands ranging in wavelength from 0.4 µm to 14.4 µm and at varying spatial resolutions (250 m, 500 m and 1 km). The Global MODIS vegetation indices (code MOD13Q1) are designed to provide consistent spatial and temporal comparisons of vegetation conditions. Red and near-infrared reflectances, centred at 645 nm and 858 nm, respectively, are used to determine the daily vegetation indices, including the well known NDVI. This index is calculated by contrasting intense chlorophyll pigment absorption in the red against the high reflectance of leaf mesophyll in the near infrared. It is a proxy of plant photosynthetic activity and has been found to be highly related to green leaf area index (LAI) and to the fraction of photosynthetically active radiation absorbed by vegetation (FAPAR). Past studies have demonstrated the potential of using NDVI data to study vegetation dynamics. More recently, several applications have been developed using MODIS NDVI data such as land-cover change detection, monitoring forest phenophases, modelling wheat yield, and other applications in forest and agricultural sciences. However, the utility of the MODIS NDVI data products is limited by the availability of high-quality data (e.g. cloud-free), and several processing steps are required before using the data: acquisition via web facilities, re-projection from the native sinusoidal projection to a standard latitude-longitude format, eventually the mosaicking of two or more tiles into a single tile. A number of processing techniques to 'smooth' the data and obtain a cleaned (no clouds) time series of NDVI imagery have also been implemented. These kind of processes are usually based on a set of ancillary information on the data quality of each pixel that are provided together with MODIS NDVI.

Set up raster time series into the database

Import MODIS NDVI time series

> raster2pgsql.exe -C -r -t 128x128 -F -M -R -N -3000 C:/modis/MOD*.tif demo_florida.ndvi_modis | psql.exe -d eurodeer_db -U postgres -p 5432

Meaning of raster2pgsql parameters

- -R: out of db raster
- -F: add a column with the name of the file
- -N: set the null value

Create and fill a field to explicitly mark the reference date of the images

Structure of the name of the original file: *MCD13Q1.A2005003.005.250m_7_days_NDVI.REFMIDw.tif*

```
ALTER TABLE demo_florida.ndvi_modis ADD COLUMN acquisition_date date;
UPDATE
  demo_florida.ndvi_modis
SET
  acquisition_date = to_date(substring(filename FROM 10 FOR 7), 'YYYYDDD');

CREATE INDEX ndvi_modis_referemce_date_index
  ON demo_florida.ndvi_modis
  USING btree
  (acquisition_date);
```

Create a table from an existing DB layer with a larger - MODIS NDVI

```
CREATE TABLE demo_florida.modis_ndvi(
  rid serial PRIMARY KEY,
  rast raster,
  filename text,
  acquisition_date date);

INSERT INTO demo_florida.modis_ndvi (rast, filename, acquisition_date)
SELECT
  rast,
  filename,
  acquisition_date
FROM
  env_data_ts.ndvi_modis_boku,
  main.study_areas
WHERE
  st_intersects(rast, ST_Expand(geom, 0.05)) AND
  study_areas_id = 1;

SELECT AddRasterConstraints('demo_florida'::name, 'modis_ndvi'::NAME, 'rast'::name);

CREATE INDEX modis_ndvi_rast_idx
  ON demo_florida.modis_ndvi
  USING GIST (ST_ConvexHull(rast));

CREATE INDEX modis_ndvi_referemce_date_index
```

```
  ON demo_florida.modis_ndvi
  USING btree
  (acquisition_date);
```

Data analysis

Extraction of a NDVI value for a point/time

```
WITH pointintime AS
(
    SELECT
      ST_SetSRID(ST_MakePoint(11.1, 46.1), 4326) AS geom,
      '2005-01-03'::date AS reference_date
)
SELECT
  ST_Value(rast, geom) * 0.0048 -0.2 AS ndvi
FROM
  demo_florida.modis_ndvi,
  pointintime
WHERE
  ST_Intersects(geom, rast) AND
  modis_ndvi.acquisition_date = pointintime.reference_date;
```

Extraction of a NDVI time series of values of a given fix

```
SELECT
  ST_X(geom) AS x,
  ST_Y(geom) AS y,
  acquisition_date,
  ST_Value(rast, geom) * 0.0048 -0.2 AS ndvi
FROM
  demo_florida.modis_ndvi,
  demo_florida.gps_data_animals
WHERE
  ST_Intersects(geom, rast) AND
  gps_data_animals_id = 1
ORDER BY
  acquisition_date;
```

Extraction of the NDVI value for a fix as temporal interpolation of the 2 closest images

```
SELECT
  gps_data_animals_id,
  acquisition_time,
  DATE_TRUNC('week', acquisition_time::date)::date,
  (trunc(
    (
    ST_VALUE(pre.rast, geom) *
    (DATE_TRUNC('week', acquisition_time::date + 7)::date - acquisition_time::date)::integer
    +
    ST_VALUE(post.rast, geom) *
    (acquisition_time::date - DATE_TRUNC('week', acquisition_time::date)::date))::integer/7)
    ) * 0.0048 -0.2
FROM
  demo_florida.gps_data_animals,
  demo_florida.modis_ndvi AS pre,
  demo_florida.modis_ndvi AS post
WHERE
  ST_INTERSECTS(geom, pre.rast) AND
  ST_INTERSECTS(geom, post.rast) AND
  DATE_TRUNC('week', acquisition_time::date)::date = pre.acquisition_date AND
  DATE_TRUNC('week', acquisition_time::date + 7)::date = post.acquisition_date AND
  gps_validity_code = 1 AND
  gps_data_animals_id = 2;
```

Extraction of the NDVI values for a set of fixes as temporal interpolation of the 2 closest images for animal 782

```
SELECT
  gps_data_animals_id,
  ST_X(geom)::numeric (8,5) AS x,
  ST_Y(geom)::numeric (8,5) AS y,
  acquisition_time,
  DATE_TRUNC('week', acquisition_time::date)::date,
  (trunc(
    (
    ST_VALUE(pre.rast, geom) *
    (DATE_TRUNC('week', acquisition_time::date + 7)::date - acquisition_time::date)::integer
    +
    ST_VALUE(post.rast, geom) *
    (acquisition_time::date - DATE_TRUNC('week', acquisition_time::date)::date))::integer/7)
    ) * 0.0048 -0.2
FROM
  demo_florida.gps_data_animals,
```

```
  demo_florida.modis_ndvi AS pre,
  demo_florida.modis_ndvi AS post
WHERE
  ST_INTERSECTS(geom, pre.rast) AND
  ST_INTERSECTS(geom, post.rast) AND
  DATE_TRUNC('week', acquisition_time::date)::date = pre.acquisition_date AND
  DATE_TRUNC('week', acquisition_time::date + 7)::date = post.acquisition_date AND
  gps_validity_code = 1 AND
  animals_id = 782
ORDER by
  acquisition_time;
```

Calculate average, max and min NDVI for the minimum convex hull of a every month for animal 782

```
SELECT
  months,
  (stats).mean  * 0.0048 - 0.2 AS ndvi_avg,
  (stats).min * 0.0048 - 0.2 AS ndvi_min,
  (stats).max * 0.0048 - 0.2 AS ndvi_max
FROM
(
  SELECT
    months,
    ST_SummaryStats(ST_UNION(ST_CLIP(rast,geom), 'max'))  AS stats
  FROM
    demo_florida.view_convexhull_monthly,
    demo_florida.modis_ndvi
  WHERE
    ST_INTERSECTS (rast, geom) AND
    EXTRACT(month FROM acquisition_date) = months AND
    months IN (1,2,3)
  GROUP BY months
  ORDER BY months
) a;
```

Plot raster time series stored in PostgreSQL/PostGIS from R

See R code