Landsat 8 Modelling Methods

Jody Holland

Contents

Introduction

The first methodological step is to train a variety of models predicting LST using LandSat 8 at 30m resolution. This is in a similar process to Onačillová et al. (2022); Equere et al.

(2021); Son et al. (2017); Karyati et al. (2022). In this modelling process we employ three types of modelling approaches:

- Linear Regression (Onačillová et al. 2022)
- Advanced Neural Networks (Equere et al. 2021)
- Geographically Weighted Regression

Through comparing the accuracy of these models we make inferences on which is most likely to provide high utility data for use in modelling 10m resolution LST with Sentinel 2 data as an input.

Libraries

This methodology will use the following r packages for data wrangling using data from July 24th 2023 (a day when both Sentinel and Landsat imagery coincides).

- The tidyverse family of packages (Wickham et al. 2019), which includes
  - dplyr for wrangling data (Wickham et al. 2023)
  - tidyr for managing data sets (Wickham, Vaughan, and Girlich 2024)
  - ggplot2 for visualisations (Wickham 2016)
- The terra package for handling raster data (Hijmans 2024)
- The RColorBrewer package for colour scales (Neuwirth 2022)
- The sf package for handling shapefile data (Pebesma and Bivand 2023)
- The leaflet package for interactive visualisation (Cheng et al. 2023)
- The osmdata package for extracting coastline data (Massicotte and South 2023)
- The car package for analysing regression results (Fox and Weisberg 2019)
- The elevatr package for extracting topological data

```r
# load packages
library(tidyverse)
library(terra)
library(RColorBrewer)
library(sf)
library(leaflet)
library(osmdata)
library(car)
```

Preparing Data

To train these models, we use input data from Landsat covering and clipped to our study area of Maspalomas-Playa del Inglés. This raster data is broken down into broken down into bands from which the following can be calculated:

- LST
- NDVI (normalised difference vegetation index)
- NDBI (normalised difference built-up index)
- NDWI (normalised difference water index)
- NDSI (normalised difference sand index)
- Surface Albedo

We will also use the following auxiliary data

- Road Proximity (from OSM)
- Proximity to Coast/Ocean
- Elevation

Loading Rasters

These rasts denote different bands of EM radiation captured by the LandSat 8 satellite over the south of Gran Canaria at 11:29am on the 24th July 2023 (U. S. Geological Surey 2023). These bands cover most of the visible light spectrum as well as the infrared spectrum. Combined in various was they can be used to calculate various measures such as the NDVI (normalised difference vegetation index) and the LST (land surface temperature).

First lets plot and load the rasts

```
# function to plot raster with a label and custome colour scale
plot_rast = function(raster, label, colors) {
  terra::plot(raster, col=colorRampPalette(colors)(100))
  mtext(text=label, side=3, line=2)
}

# define the bounding box epsg:4083
bbox = ext(c(437807.3121, 461489.5356, 3067557.7221, 3081455.2224))

# load each raster, crop to bounding box, and plot with label
b1 = crop(rast("landsat/B1.TIF"), bbox)
b2 = crop(rast("landsat/B2.TIF"), bbox)
b3 = crop(rast("landsat/B3.TIF"), bbox)
```

```
b4 = crop(rast("landsat/B4.TIF"), bbox)
b5 = crop(rast("landsat/B5.TIF"), bbox)
b6 = crop(rast("landsat/B6.TIF"), bbox)
b7 = crop(rast("landsat/B7.TIF"), bbox)
b10 = crop(rast("landsat/B10.TIF"), bbox)
```

Loading Coast Shapefile

Using OSM Data to first extract a coastline of the island, this process enables such analysis as calculating distance from the coastline

```
# query osm
gran_canaria_query = opq(bbox = "Gran Canaria") %>%
  add_osm_feature(key = "place", value = "island") %>%
  osmdata_sf()
# extract the coast data
gran_canaria_sf = gran_canaria_query$osm_multipolygons

# make spatial vector
gran_canaria_vect = st_transform(gran_canaria_sf, crs(b1)) %>%
  vect()
```

Loading Road Shapefile

Again using OSM data, it is possible to extract road data in the form of a spatial vector.

```
# define bbox
bbox_osm = c(-15.630970,27.730945,-15.391159,27.857290)

# query osm
osm_roads = opq(bbox = bbox_osm) %>%
  add_osm_feature(key = "highway") %>%
  osmdata_sf()

# extract road lines
roads_vect = st_transform(osm_roads$osm_lines, crs(b1)) %>%
  vect()
```

Masking Ocean Values

```r
b1 = mask(b1, gran_canaria_vect)
b2 = mask(b2, gran_canaria_vect)
b3 = mask(b3, gran_canaria_vect)
b4 = mask(b4, gran_canaria_vect)
b5 = mask(b5, gran_canaria_vect)
b6 = mask(b6, gran_canaria_vect)
b7 = mask(b7, gran_canaria_vect)
b10 = mask(b10, gran_canaria_vect)
```

Calculating NDVI

Perhaps the most important metric beside LST is NVDI. NDVI is also used in the computation of impassivity and ultimately therefore emissivity adjusted LST.

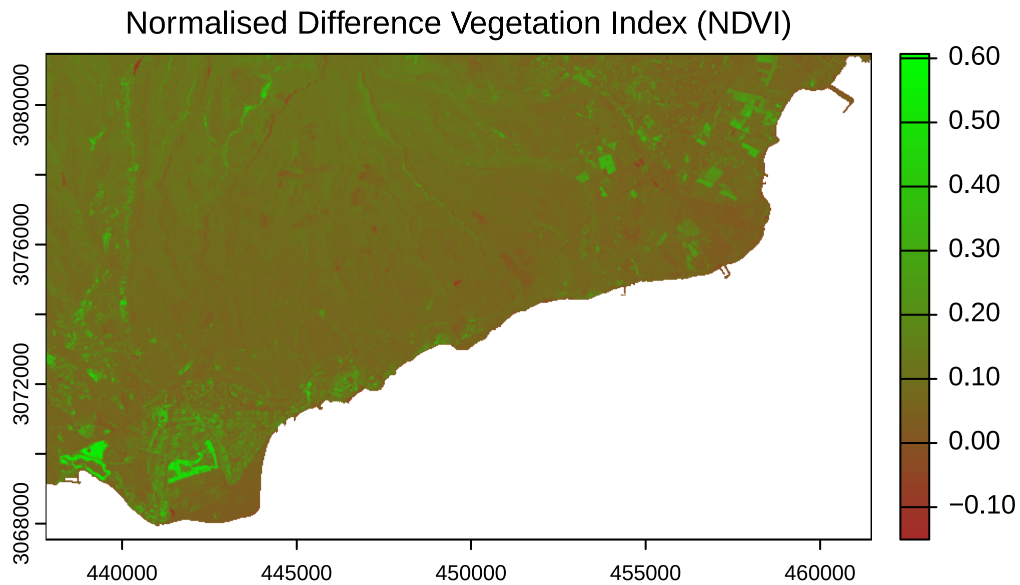
To calculate NDVI, the following formula is used:

$$\text{NDVI} = \frac{\text{NIR} - \text{Red}}{\text{NIR} + \text{Red}}$$

Here NIR represents the Near-Infrared band reflectance value, in Landsat 8 this is Band 5. Further *Red* represents the Red band reflectance value, in Landsat 8 this is Band 4.

```r
# calc and plot ndvi using bands 5 and 4
ndvi = (b5 - b4) / (b5 + b4)

plot_rast(ndvi,
          "Normalised Difference Vegetation Index (NDVI)",
          c("brown", "green"))
```



Calculating NDBI

Another important metric is NDBI as urban artificial surfaces may trap heat causing excess heating relative to vegetated surfaces.

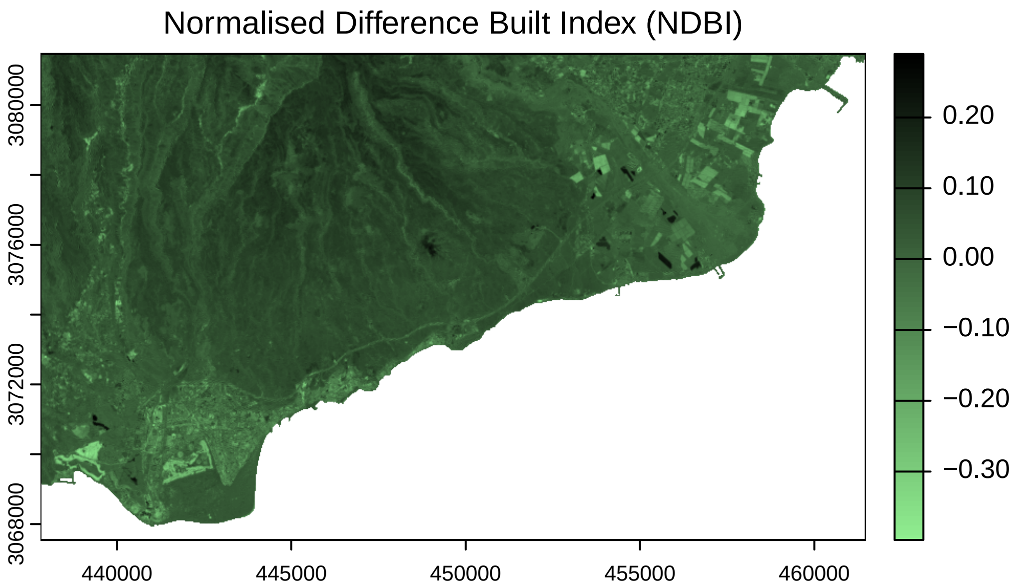
To calculate NDBI for Landsat 8, the following formula is used (Abulibdeh 2021):

$$\text{NDVI} = \frac{\text{MIR} - \text{NIR}}{\text{MIR} + \text{NIR}}$$

Here NIR again represents the Middle-Infrared band value, in Landsat 8 this is Band 5. Also NIR again represents the Near-Infrared band value, being Landsat 8 Band 5.

```
ndbi = (b6 - b5) / (b6 + b5)

plot_rast(ndbi,
          "Normalised Difference Built Index (NDBI)",
          c("lightgreen", "black"))
```



Calculating NDWI

Complementing these metrics is NDWI, here used to analyse the impact that surface water has upon the temperature.

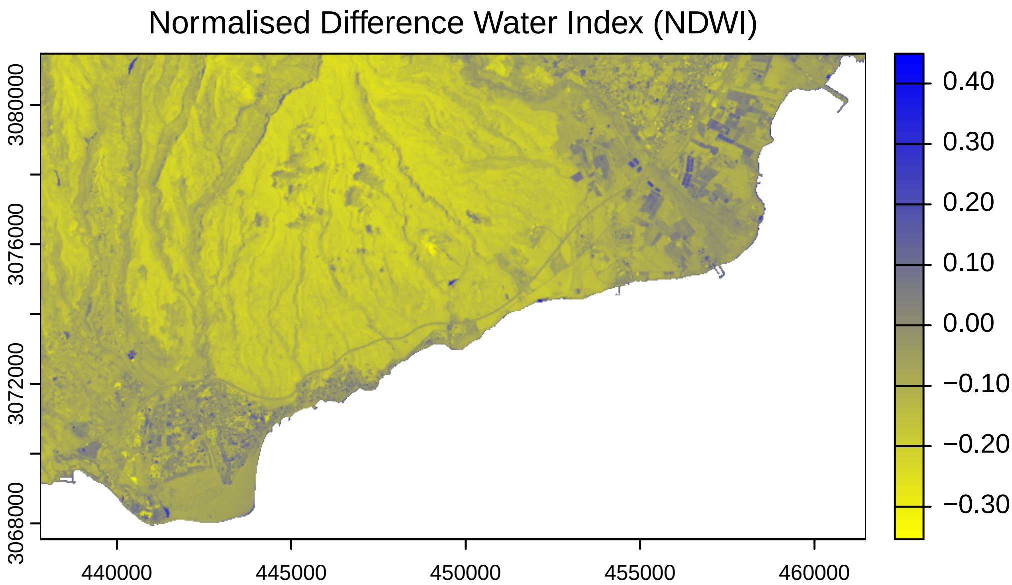
To calculate NWBI for Landsat 8, the following formula is used:

$$\text{NDVI} = \frac{\text{Green} - \text{SWIR}}{\text{Green} + \text{SWIR}}$$

Here $Green$ represents the Green reflectance values, in Landsat 8 this is Band 3. Also $SWIR$ again represents the Short-Infrared band value, being Landsat 8 Band 7.

```
# calc and plot
ndwi = (b3 - b7) / (b3 + b7)

plot_rast(ndwi,
          "Normalised Difference Water Index (NDWI)",
          c("yellow", "blue"))
```



Calculating Surface Albedo

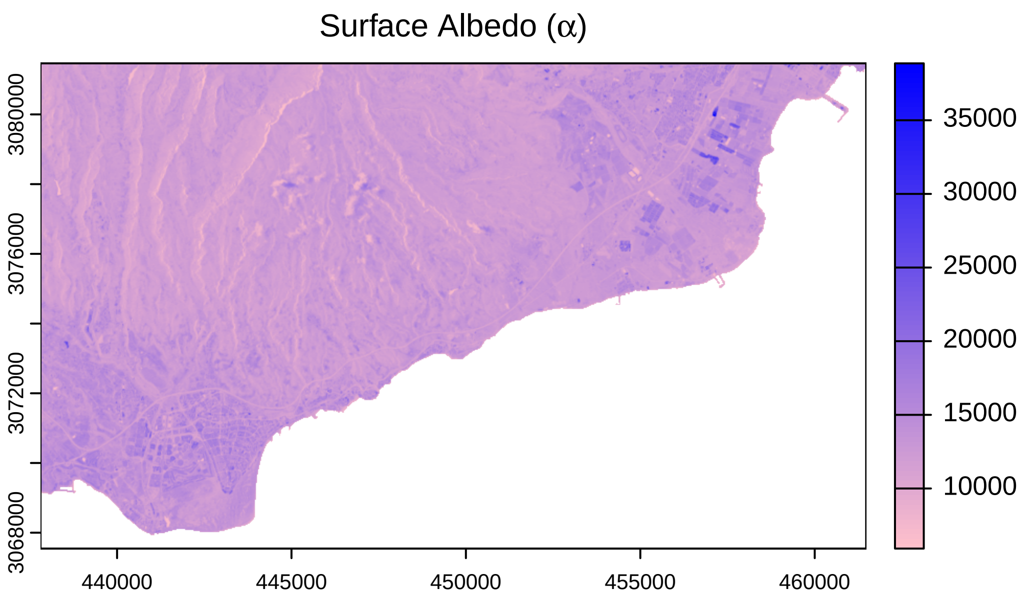
There is also surface albedo α, which is the the fraction of the reflected shortwave radiation, is calculated using the following formula (Equere et al. 2021):

$$\alpha = \frac{0.356 \cdot \text{Blue} + 0.0130 \cdot \text{Red} + 0.373 \cdot \text{NIR} + 0.085 * \text{MIR} + 0.072 \cdot \text{SWIR} - 0.0018}{1.016}$$

Here *Blue* represents the Blue reflectance values, in Landsat 8 this is Band 2. *Red* again represents the Red band value, being Landsat 8 Band 4. NIR represents the Near-Infrared values, in Landsat 8 this is Band 5. MIR likewise represents the Middle-Infrared values, being Landsat 8 Band 6. Finally $SWIR$ represents the Near-Infrared values, in Landsat 8 this is Band 7.

```
# calc and plot
albedo = (0.356 * b2 + 0.0130 * b4 + 0.373 * b5 + 0.085 * b6 + 0.072 * b6 - 0.0018) / 1

plot_rast(albedo,
          expression(paste("Surface Albedo (", alpha,")")),
          c("pink", "blue"))
```



Calculating Distance from Ocean

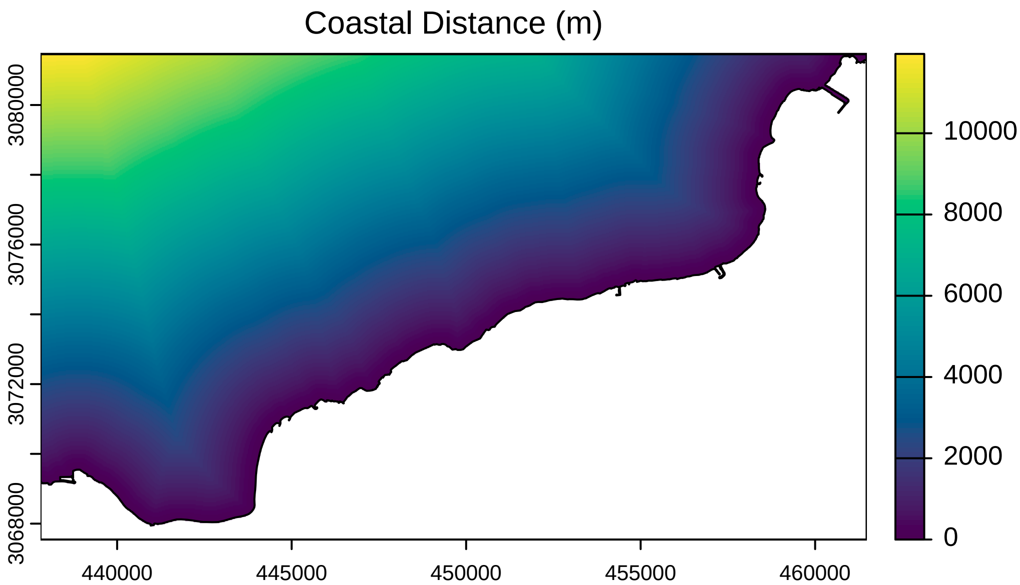
In terms of other environmental variables, there is also distance from the coast to consider, calculated using OSM coastline data.

```
# create blank template
template = crop(rast("landsat/B1.TIF"), bbox)

# crop sp by bbox
gran_canaria_vect = crop(gran_canaria_vect, bbox)

# create land raster
sea_mask = rasterize(gran_canaria_vect,
                     template,
                     NA,
                     background=1)


# calc distance and plot
coastdistance = distance(sea_mask) %>%
  mask(gran_canaria_vect)
terra::plot(coastdistance, col = hcl.colors(100))
plot(gran_canaria_vect, add = TRUE)
mtext(text="Coastal Distance (m)", side=3, line=2)
```

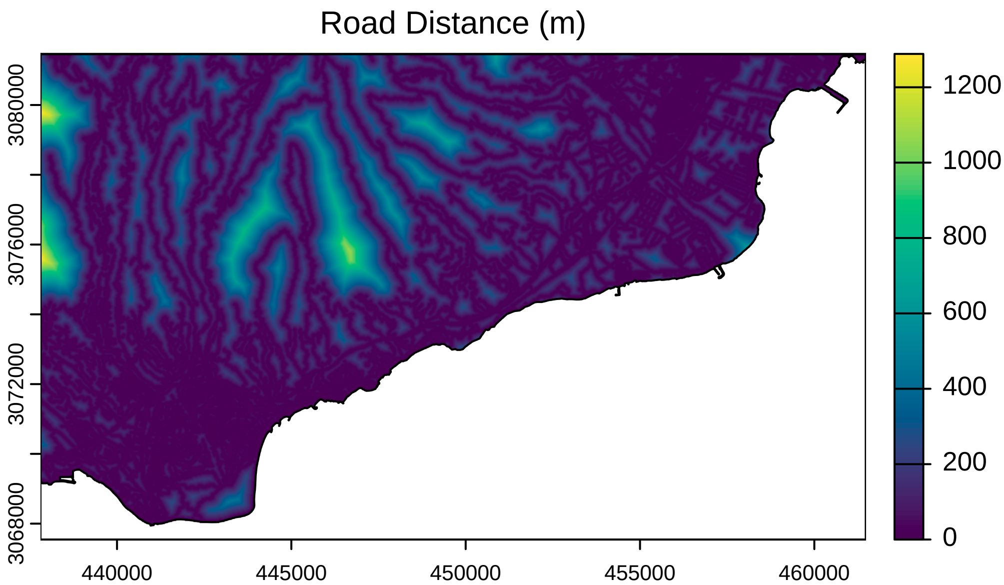


Calculating Distance from Roads

Furthermore, there is distance from the nearest road, again calculated using OSM coastline data.

```
# create road raster
road_mask = rasterize(roads_vect,
                      template,
                      1,
                      background=NA)

# calc distance and plot
roaddistance = distance(road_mask) %>%
  mask(gran_canaria_vect)
plot(roaddistance, col = hcl.colors(100))
plot(gran_canaria_vect, add = TRUE)
mtext(text="Road Distance (m)", side=3, line=2)
```



Calculating Elevation

Calculating LST

Calculating LST can be straightforward in principle, using Band 10 alone of Landsat 8 data. However, this will not account for surface emissivity. Thus, for this study, an adjusted LST is calculated.

Top of Atmosphere Spectral Radiance

The first step is to convert the Digital Numbers DN in the Band 10 layer to Top of Atmosphere Spectral Radiance L_λ values. This process uses input units found the images metadata/MLT file. These values are the RADIANCE_MULT_BAND_10 M_L and the RADIANCE_ADD_BAND_10 A_L.

Putting these together into a formula is as follows:

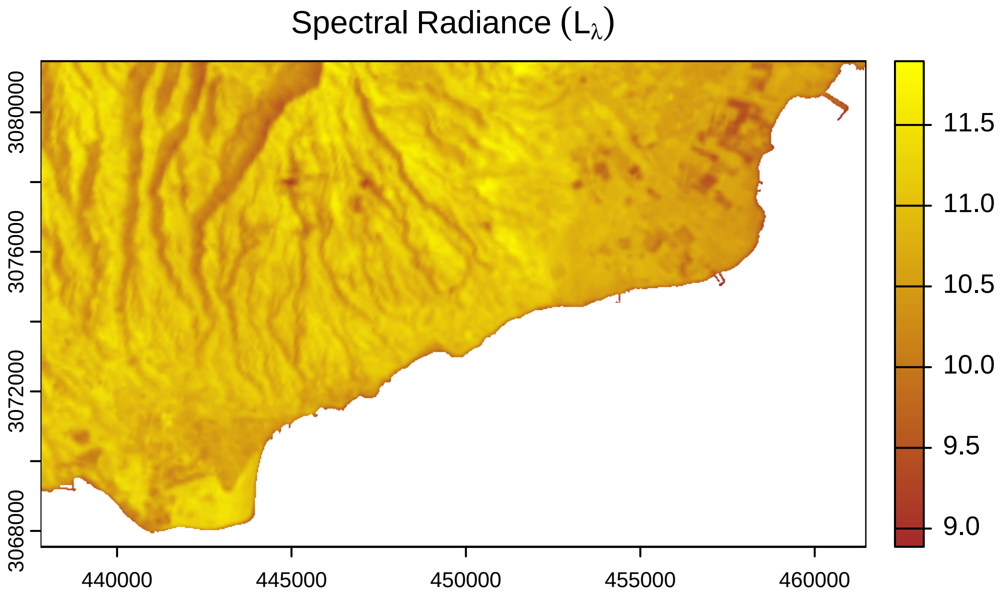
$$\mathrm{L}_\lambda = \mathrm{M}_L \cdot \mathrm{DN} + \mathrm{A}_L$$

```
# define metadata values
mult_b10 = 3.8e-04
add_b10 = 0.1

# calc and plot
```

```
l = (mult_b10 * b10 + add_b10)

plot_rast(l,
          expression(Spectral~Radiance~(L[lambda])),
          c("brown", "yellow"))
```



At-Sensor Temperature

The second step is "Conversion of Radiance to At-Sensor Temperature" (Karyati et al. 2022). This This converts the Spectral Radiance into temperature values or Brightness Temperature BT in Kelvin. This process requires two constants related to Landsat 8 Band 10's specific thermal conversion constants found again in the metadata file. These are $K1$and $K2$.

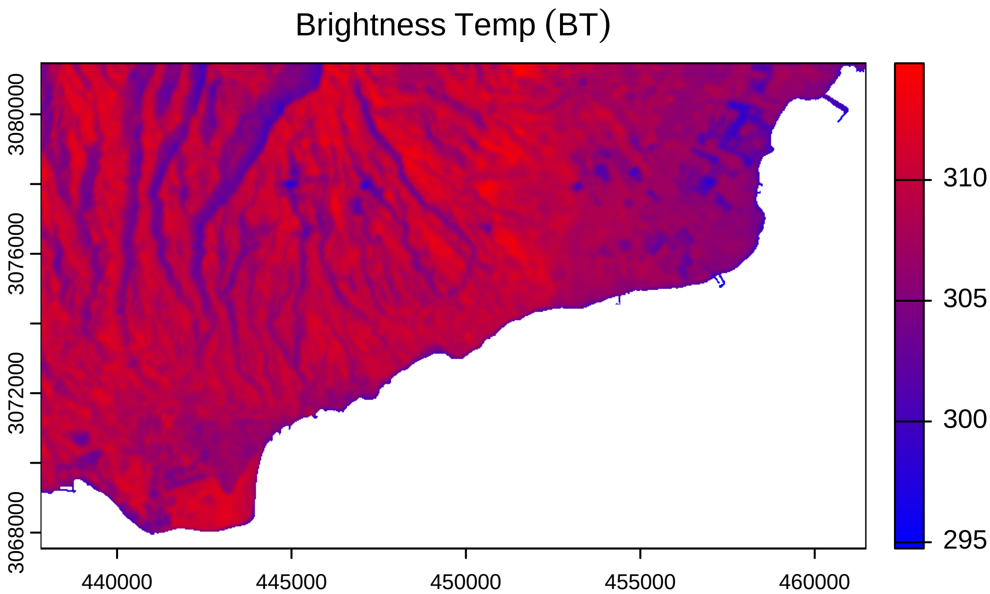
Putting this together into a formula is as follows:

$$\mathrm{BT} = \frac{\mathrm{K2}}{ln(\frac{\mathrm{K1}}{\mathrm{L}_\lambda} + 1)}$$

```
# add effective wavelength of landsat b10
k1 = 799.0284
k2 = 1329.2405
```

```
# calc and plot
bt = (k2 / log(k1 / l + 1))

plot_rast(bt,
          expression(Brightness~Temp~(BT)),
          c("blue", "red"))
```



Surface Level Emissivity

The above metric alone could be used for study of heat distribution, however it does not account for the variation caused by surface level emissivity. Emissivity denotes how much infrared a surface will absorb or reflect, ranging from a mirror to a perfect black surface. The values of emissivity range from 0 to 1 respectively.

Fractional Vegetation Factor

NDVI is used to determine emissivity, through the fractional vegetation factor P_v. This value is specific to the region being studied.

The formula for calculating fractional vegetation factor is as follows:

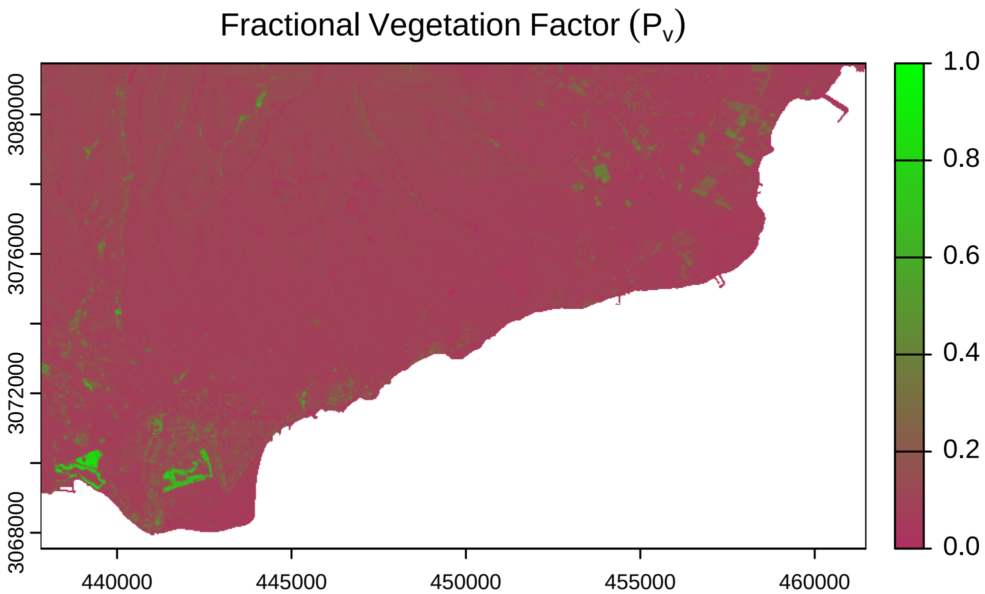
$$\mathrm{P}_v = \left(\frac{\mathrm{NDVI} - \mathrm{NDVI}_{\mathrm{min}}}{\mathrm{NDVI}_{\mathrm{max}} - \mathrm{NDVI}_{\mathrm{min}}}\right)^2$$

Using minimum and maximum values relative to the area.

```r
# define min and max ndvi
min_ndvi = global(ndvi, fun=min, na.rm=TRUE) %>%
  as.numeric()
max_ndvi = global(ndvi, fun=max, na.rm=TRUE) %>%
  as.numeric()

# calc and plot vegetation factor
pv = ((ndvi - min_ndvi) / (max_ndvi - min_ndvi))^2

plot_rast(pv,
          expression(Fractional~Vegetation~Factor~(P[v])),
          c("maroon", "green"))
```



Emissivity

From this the surface emissivity can be calculated. This uses two constants relating to the relative emissivity of soil $\epsilon s\lambda$ and the emissivity of vegetation $\epsilon v\lambda$, as well as one surface roughness $C\lambda$ metric of 0.005

The formula for estimating emissivity $\epsilon\lambda$ is thus as follows:

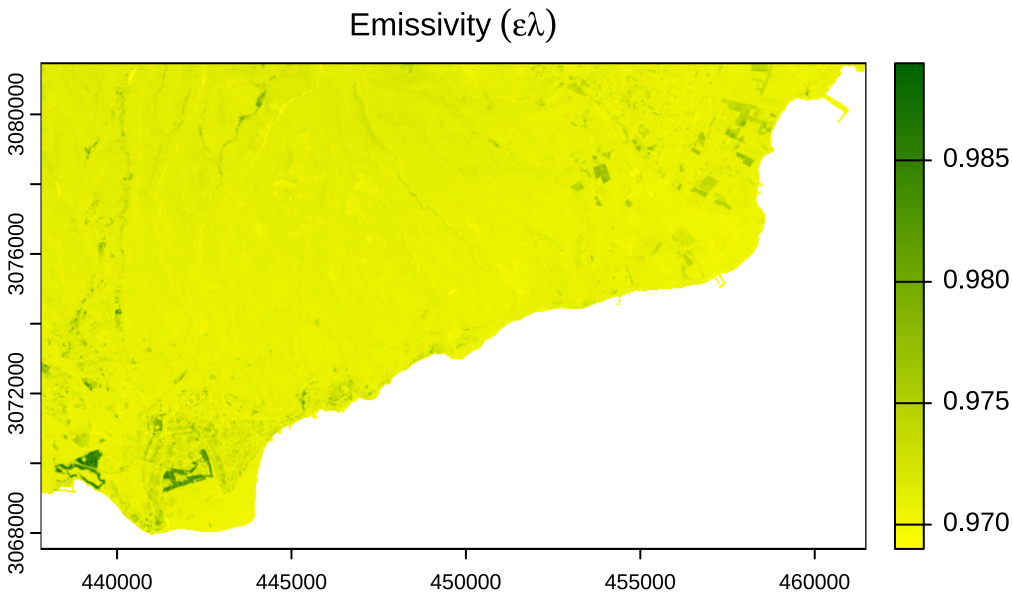
$$\epsilon\lambda = \epsilon v\lambda + \epsilon s\lambda \cdot (1 - P_v) + C\lambda$$

```
# soil emissivity
es = 0.964

# veg emissivity
ev = 0.984

# calc and plot emissivity
e =  ev * pv + es * (1 - pv) + 0.005

plot_rast(e,
          expression(Emissivity~(epsilon*lambda)),
          c("yellow", "darkgreen"))
```



Correction Constant

Now most of the building blocks are in place to calculate the adjusted LST. This requires a correction constant ρ that incorporates the Speed of Light c, Planck's constant h, and Boltzmann's constant σ.

The formula for this is as follows:

$$\rho = \frac{c \cdot h}{\sigma}$$

```r
# speed of light
c = 2.997925e8

# planck's constant
h = 6.626070e-34

# boltzmann's constant
sigma = 1.380649e-23

# calc and output correction constant/rho
rho = c * h / sigma
rho
```

```
[1] 0.01438777
```

Land Surface Temperature

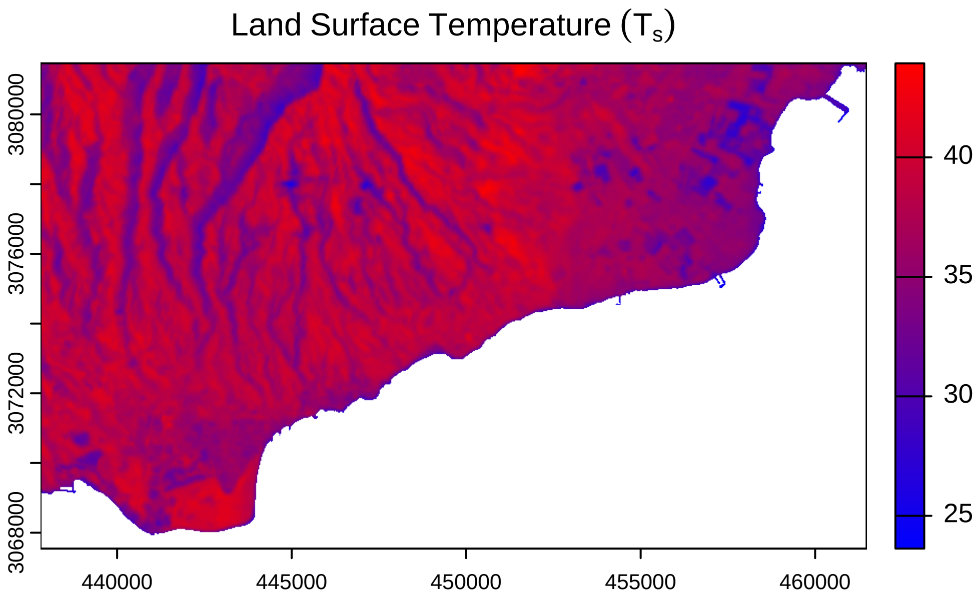
With these building blocks, plus the wavelength of the emitted radiance for Landsat 8 λ, the adjusted LST Ts can be determined using the following formula (converting from Kelvin to Celcius:

$$\mathrm{T}_s = \frac{\mathrm{BT}}{1 + (\frac{\lambda \mathrm{BT}}{\rho} \cdot ln(\epsilon\lambda))} - 273.15$$

```r
# wavelength
wl = 10.895e-6

# calc and plot
ts = bt / (1 + (wl * bt /  rho) * log(e)) - 273.15

plot_rast(ts,
          expression(Land~Surface~Temperature~(T[s])), c("blue", "red"))
```



Leaflet Visualisation of LST

Using the Leaflet Package for R (Cheng et al. 2023), we can visualise the resulting rast

```
plet(ts,
     col=(c("darkblue","pink")),
     legend="bottomright",
     main="LST (Cº)",
     tiles=c("Esri.WorldImagery"))
```

Model Building

```
rast_stack = c(ts, ndvi, ndbi, ndwi, albedo, coastdistance, roaddistance)

data_df = na.omit(as.data.frame(rast_stack, xy=TRUE))

names(data_df) = c("X", "Y", "LST", "NDVI", "NDBI", "NDWI", "Albedo", "CoastDistance", "

model = lm(LST ~ NDVI + NDBI + NDWI  + Albedo + CoastDistance + RoadDistance, data=data_

summary(model)
```

```
Call:
lm(formula = LST ~ NDVI + NDBI + NDWI + Albedo + CoastDistance +
    RoadDistance, data = data_df)

Residuals:
     Min       1Q   Median       3Q      Max
-14.8802  -1.1699   0.0778   1.2823   9.7995

Coefficients:
                Estimate Std. Error  t value Pr(>|t|)
(Intercept)    3.105e+01  4.457e-02  696.740   <2e-16 ***
NDVI          -9.508e+00  1.620e-01  -58.690   <2e-16 ***
NDBI          -2.759e+00  2.467e-01  -11.186   <2e-16 ***
NDWI          -2.296e+01  1.631e-01 -140.802   <2e-16 ***
Albedo         3.483e-04  3.368e-06  103.403   <2e-16 ***
CoastDistance  1.665e-05  2.008e-06    8.292   <2e-16 ***
RoadDistance  -1.570e-03  2.868e-05  -54.734   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.002 on 220228 degrees of freedom
Multiple R-squared:  0.3622,    Adjusted R-squared:  0.3622
F-statistic: 2.084e+04 on 6 and 220228 DF,  p-value: < 2.2e-16
```

Abulibdeh, Ammar. 2021. "Analysis of Urban Heat Island Characteristics and Mitigation Strategies for Eight Arid and Semi-Arid Gulf Region Cities." *Environmental Earth Sciences* 80 (7): 259. https://doi.org/10.1007/s12665-021-09540-7.

Cheng, Joe, Barret Schloerke, Bhaskar Karambelkar, and Yihui Xie. 2023. "Leaflet: Create Interactive Web Maps with the JavaScript 'Leaflet' Library." https://CRAN.R-project.org/package=leaflet.

Equere, Victor, Parham A. Mirzaei, Saffa Riffat, and Yilin Wang. 2021. "Integration of Topological Aspect of City Terrains to Predict the Spatial Distribution of Urban Heat Island Using GIS and ANN." *Sustainable Cities and Society* 69 (June): 102825. https://doi.org/10.1016/j.scs.2021.102825.

Fox, John, and Sanford Weisberg. 2019. "An {r} Companion to Applied Regression." https://socialsciences.mcmaster.ca/jfox/Books/Companion/.

Hijmans, Robert J. 2024. "Terra: Spatial Data Analysis." https://CRAN.R-project.org/package=terra.

Karyati, N E, R I Sholihah, D R Panuju, B H Trisasongko, D Nadalia, and L O S Iman. 2022. "Application of Landsat-8 OLI/TIRS to Assess the Urban Heat Island (UHI)." *IOP Conference Series: Earth and Environmental Science* 1109 (1): 012069. https://doi.org/10.1088/1755-1315/1109/1/012069.

Massicotte, Philippe, and Andy South. 2023. "Rnaturalearth: World Map Data from Natural Earth." https://CRAN.R-project.org/package=rnaturalearth.

Neuwirth, Erich. 2022. "RColorBrewer: ColorBrewer Palettes." https://CRAN.R-project.org/package=RColorBrewer.

Onačillová, Katarína, Michal Gallay, Daniel Paluba, Anna Péliová, Ondrej Tokarčík, and Daniela Laubertová. 2022. "Combining Landsat 8 and Sentinel-2 Data in Google Earth Engine to Derive Higher Resolution Land Surface Temperature Maps in Urban Environment." *Remote Sensing* 14 (16): 4076. https://doi.org/10.3390/rs14164076.

Pebesma, Edzer, and Roger Bivand. 2023. "{Spatial Data Science: With Applications in r}." https://doi.org/10.1201/9780429459016.

Son, Nguyen-Thanh, Chi-Farn Chen, Cheng-Ru Chen, Bui-Xuan Thanh, and Tran-Hau Vuong. 2017. "Assessment of Urbanization and Urban Heat Islands in Ho Chi Minh City, Vietnam Using Landsat Data." *Sustainable Cities and Society* 30 (April): 150–61. https://doi.org/10.1016/j.scs.2017.01.009.

U. S. Geological Surey. 2023. "Landsat - 8 Image."

Wickham, Hadley. 2016. "Ggplot2: Elegant Graphics for Data Analysis." https://ggplot2.tidyverse.org.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the {Tidyverse}" 4: 1686. https://doi.org/10.21105/joss.01686.

Wickham, Hadley, Romain François, Lionel Henry, Kirill Müller, and Davis Vaughan. 2023. "Dplyr: A Grammar of Data Manipulation." https://CRAN.R-project.org/package=dplyr.

Wickham, Hadley, Davis Vaughan, and Maximilian Girlich. 2024. "Tidyr: Tidy Messy Data." https://CRAN.R-project.org/package=tidyr.