Assessment of the surface urban heat island in Bern by fusing the Landsat and MODIS LST products

Master thesis
Faculty of Science, University of Bern

handed in by
Thierry Benjamin Müller

2023

Supervisor:
Prof. Dr. Stefan Brönnimann

-

Co-Supervisor:
Prof. Dr. Stefan Wunderle

-

Advisor:
Dr. Moritz Gubler

Content

Abstract

With rising temperatures due to climate change and more people living in urbanised areas worldwide, the urban heat island effect has become an important topic in the public and scientific discourse. The different surface properties of a city compared to rural areas lead to more heat uptake and increased temperatures in built-up areas, especially during summer. When looking at surface temperatures in isolation, the effect of higher temperatures in urban areas is called surface urban heat island (SUHI). To measure surface temperatures over a large area, thermal satellite data are most commonly used. However, there is always a trade-off between spatial and temporal resolution for a single sensor. Landsat has a high spatial resolution, while Modis has a high temporal resolution. This study uses the fusion-based downscaling method “ESTARFM”, to combine Landsat and MODIS data to improve spatiotemporal resolution by generating synthetic Landsat images. The ESTARFM method is able to generate accurate surface temperature maps with average RMSE values of 1.811K compared to original Landsat images. The most significant limitations of this method are the following: it can only be used for cloud-free images, and areas undergoing change in surface properties during the study period, for example agricultural areas with changing vegetation, show high errors compared to other sites. After validation and accuracy assessments, the synthetic Landsat images are used to study surface temperature distributions and the SUHI effect in the city of Bern, Switzerland, in the summer of 2019. The results show that the SUHI effect is present in Bern, but it varies between 0K and 16K depending on the location within the city. The SUHI maps also show the significant heating effect of sealed surfaces, while vegetated areas within the city show lower surface temperatures.

1 Introduction

Human-induced climate change, including more frequent and intense extreme events, has caused widespread adverse impacts and related losses and damage to nature and people beyond natural climate variability (IPCC 2022, 2022a). Especially in cities and urban areas, the risk faced by people and assets from hazards associated with climate change has increased. This problem is amplified by the fact that the number of people expected to live in urban areas highly exposed to climate change impacts has grown substantially (IPCC 2022, 2022b). In Switzerland, increasing temperatures are one of the most noticeable impacts of climate change, with the seven hottest years since the start of the measurements occurring after the year 2010 (MeteoSchweiz, 2022b). Since 1864, the boundary layer air temperatures have increased by 2.1°C on average, which is almost double compared to the global mean of 1.1°C. An increase in mean temperature also brings stronger extreme events in the form of heat waves (MeteoSchweiz, 2021). In Europe, heat and heat waves are the biggest weather- and climate-related threat to human health. The latest example is the very hot summer of 2022, where a large number of excess deaths all over Europe could be attributed to extreme temperatures and prolonged heat waves. In the coming decades, this problem will grow further as heatwaves are expected to happen more frequently with longer durations and higher intensity. At the same time, the population becomes more vulnerable due to the ageing population and growing urbanisation (EEA, 2022).

Such temperature extremes are exacerbated in cities due to the urban heat island (UHI) effect. The UHI describes the fact that cities often have locally higher temperatures compared to their rural surroundings (Oke et al., 2017). Most relevant for human health is the canopy layer UHI, which is the difference of temperatures in urban street canyons 1-3m above ground compared to the rural temperatures at the same height (Stewart, 2011). The reasons for these differences are physical changes in the surface energy balance caused by urban development. Urban surfaces tend to have a high heat capacity, storing more sensible heat during the day than rural surfaces. In addition, urban surfaces dry out faster and together with limited vegetation, leading to increased sensible heat fluxes while latent heat fluxes are reduced (Oke et al., 2017). Reduced wind speeds and smaller sky view factors, both caused by the 3D structure of a city, lead to an obstruction of sensible heat compared to rural areas, meaning that urban areas cool down slower after sunset. Because of these effects, the UHI is most pronounced during night-time (Stewart, 2011). A different picture can be seen when looking at the surfaces themselves. When measuring land surface temperatures (LST), the difference between urban and rural surfaces is called surface urban heat island (SUHI) (Parlow et al., 2014). Even though surface and air temperature are connected, they behave very differently. The air temperature does not necessarily follow the spatial pattern of surface temperatures, so comparing them is difficult (Parlow, 2013). Due to the different energy balances of the surface itself and the near-surface air, the resulting heat islands also differ. The magnitude of the SUHI is usually stronger during daytime compared to nighttime due to different albedo, heat conductivity and heat capacity of urban and rural surfaces (Parlow, 2013; Zhou et al., 2018).

Measuring the UHI is often challenging as it is never uniform and varies significantly within a city. Most official measurement stations for cities are often not in the city centre, where temperatures usually rise most, but in the suburbs or at the city's edge. This is also the case in Bern, where the official measurement station is situated in Zollikofen, about 5km north of the city centre, in an open field and not in the urbanised area. Therefore, it isn't easy to measure the UHI with just the official weather stations available in most places, as they are sparse and not representative for different parts of a city (Zhou *et al.*, 2018). To study the UHI effect in Bern, an air temperature measurement network has been set up, using low-cost sensors to measure air temperatures within the city and its rural surroundings. It was shown that even in small cities like Bern, the UHI effect is present and can be measured (Gubler *et al.*, 2021). However, a limitation of these in situ measurements across the city is that they are still point measurements, which are not equally spaced. Areas around and between the measurement points can only be estimated by interpolation or statistical methods. Burger et al. (2021) used land use regression (LUR) models to calculate the UHI intensity in Bern using the measurement network as validation, showing that the strongest intensities can be found in the old town, which is the most densely built-up part of the city.

Measuring LST over large areas is another challenge, as satellite data is not seamlessly available during the day. Therefore, surface temperatures over large areas are often estimated by geostatistical modelling, but it is essential to verify those models with actual thermal satellite data. The measured brightness temperatures from thermal satellite sensors can be used to calculate LST (Sobrino et al., 2018). Commonly used satellite platforms with thermal sensors like MODIS and Landsat have calculated and calibrated LST products available for free (Wan, 2013; USGS, 2022). Those are usually well-documented and have been validated thoroughly for different locations and settings (Mao et al., 2021). Like the UHI, the SUHI is defined as the temperature difference between the urban and the surrounding rural areas but for surface temperature instead of air temperature. Because surface temperatures are rarely measured in situ, distinguishing between urban and rural areas is often difficult. Thus, there are different ways to calculate the SUHI. One way is to define it as the difference between the maximum temperature in the city and a representative temperature for the surrounding rural areas (Zhou et al., 2018). Another possibility is to calculate the spatial mean temperature for both areas and subtract them. A third way is to calculate the difference spatially, where one representative rural temperature is subtracted from all other points to calculate the difference over the whole area, showing the spatial structure of the SUHI (Oke *et al.*, 2017).

When using thermal satellite data to calculate LSTs, it is important to remember that the sensors can never see the entire surface, especially when many buildings and street canyons exist. A satellite measuring over a city always sees more of the roofs than the walls of the buildings or the streets, which can influence the measured surface temperatures (Parlow et al., 2014). Remotely sensed surface temperatures are a function of the city's thermal state, the instrument's optical properties, including the position and viewing angle of the satellite and the position of the sun (Yao *et al.*, 2018). The timing of the overpass plays a role as well, which is around the same time every day for MODIS and Landsat. This makes it difficult to gain information about the development of the SUHI over the course of the day (Oke *et al.*, 2017).

Almost no recent studies in Switzerland look at the SUHI, except in Basel, where UHI and SUHI were compared using different methods (Parlow et al., 2014). A significant limitation in using LST is data availability in adequate spatial and temporal resolution, especially for heterogeneous areas like cities and their surroundings. Single satellite-based sensors cannot deliver high enough spatial and temporal resolution due to the trade-off between scanning swath and pixel size (Mao et al., 2021). To solve this problem, different downscaling techniques have been developed to enhance the spatial or temporal resolutions of satellite images. They were categorised into three main output groups: Spatial resolution enhancement, temporal resolution enhancement and simultaneous spatiotemporal resolution enhancement (Mao et al., 2021). Another categorisation would be into kernel-driven methods and fusion-based methods (Xia et al., 2019). Kernel-based methods often use auxiliary data statistically correlated with LST from the visible and near-infrared bands of the same sensor, as they usually have higher spatial resolutions than the thermal bands. One example would be the normalised vegetation index, which correlates well with the LST distribution, especially in cities (Xia *et al.*, 2019). Fusion-based methods on the other hand, use LST data from two or more thermal sensors with differing resolutions and fuse them to achieve higher resolutions. The basic idea is to have one sensor with frequent coarse resolution and one with sparse fine resolution to produce synthesised images with high spatial and temporal resolution (Wu *et al.*, 2021). Spatial resolution enhancement approaches are primarily based on kernel-driven methods, while temporal and simultaneous spatiotemporal enhancements are mostly fusion-based or a combination of both approaches (Mao et al., 2021). In this study, the focus is on fusion-based methods, with the goal to improve the temporal resolution of a sensor that already has a satisfying spatial resolution.

One of the most commonly used Fusion-based methods is called STARFM. It was introduced by Gao et al. (2006) and was first used for reflectance images, not LST. STARFM generates high spatial resolution images by using a pair of high- and low-resolution images at the reference time and a low-resolution image at the target time. The method uses a moving window to predict the high-resolution image of the target time by finding similar pixels and weighting them to calculate the centre pixel's value (Gao *et al.*, 2006). However, the STARFM and its variants show low accuracy and weak robustness in complex and heterogeneous regions (Zhu *et al.*, 2010). Therefore, many studies modified the original STARFM and improved the model performance over landscapes with heterogeneous land cover. These modified methods have also been used to fuse LST data (Mao et al., 2021). For example, Liu et al. (2012) directly applied STARFM to ASTER and MODIS thermal data. Wu P. et al. (2015) used STARFM as a basis and proposed the spatiotemporal integrated temperature fusion model, which takes LSTs from polar-orbiting and geostationary satellite observations to generate diurnal Landsat-like LSTs. Weng et al. (2014) introduced the Spatial-temporal Adaptive Data Fusion Algorithm for Temperature mapping algorithm for LST prediction based on the STARFM algorithm framework. The most significant difference to STARFM is that they considered the annual temperature cycle and thermal landscape heterogeneity. Huang et al. (2013) used the STARFM framework and added bilateral filtering to try and improve the weighting function by including the effects of neighbouring pixels on LST. They used Landsat and MODIS data and downscaled LST data in Beijing to monitor the urban thermal environment. Zhu et al. (2010) developed the enhanced STARFM (ESTARFM), which improves the basic STARFM in heterogeneous areas by introducing a conversion coefficient and using more input images.

When using fusion-based methods, the assumption is that remotely sensed data from different sensors at the same date can be compared and are correlated with each other after radiometric calibration, geometric rectification, and atmospheric correction. Nevertheless, some systemic biases are expected due to orbit parameters, slightly different acquisition times within the day and different bandwidths (Zhu *et al.*, 2010). Another possible source of bias is change in land cover between the reference dates, resulting in some uncertainty in selecting similar pixels. Therefore, the method should work better if images acquired near the prediction time are used to retrieve the unknown fine-resolution LSTs (Zhu et al., 2010).

There are no studies yet for the city of Bern that look at the SUHI using enhanced LST data. Assuming that the spatial resolution of the Landsat LST from the Collection 2 (C2) Level 2 Science Product (L2SP) is high enough to analyse the SUHI, this study aims to apply a fusion-based algorithm directly to available LST data from MODIS and Landsat 8 to generate Landsat-like LSTs to fill the gaps between the Landsat overpasses to improve the temporal resolution. While most studies applied the fusion models to TIR radiance data because the Landsat LST product is only available since the year 2020, this study looks to apply a fusion-based method directly to available LST products from MODIS and

Landsat. First, the downscaling results are validated using the available Landsat scenes. The generated Landsat-like LST maps are then used to calculate the SUHI in Bern over the whole summer of 2019 to better understand its extent, intensity and development during the summer months.

2 Data and Methods

2.1 Study Area and Period

The study area includes the city of Bern and its close surroundings. The extent of the study area where the downscaling is performed can be seen as the red rectangle in Fig. 1. It is 247 x 247 Landsat pixels large, each pixel having a spatial resolution of 30m. The centre of the study area is located at 7°25'52.619"E 46°57'45.097"N. Bern is located between the Alps and the Jura mountains, with a mean elevation of 540m above sea level. In this area, we find different types of surfaces, including forested areas, agricultural land, built-up urban areas and water bodies with the Aare flowing through the city. At the official weather station "Bern / Zollikofen", the yearly mean temperature from 1991-2020 is 9.3°C, and the mean annual precipitation sum is 1021.8 mm. Temperature and precipitation are highest during the summer months (MeteoSchweiz, 2022a). The study uses data from May 15 to September 15 2019, as the summer months are most relevant in urban heat studies. The year 2019 was chosen because there are enough available Landsat scenes during the summer and the possibility to compare the SUHI values with the UHI values from the low-cost temperature measurement network in Bern, which is available starting in 2018.

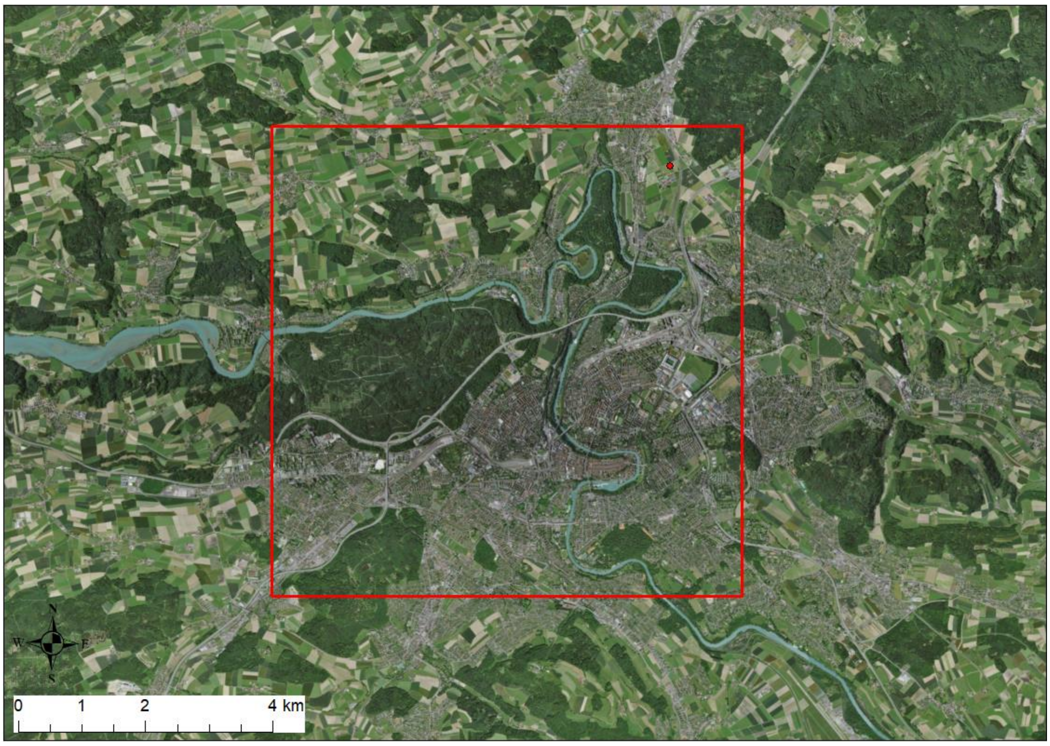


Figure 1 Landsat 8 colour composite reflectance image of Bern and its surroundings with the extent of the study area in red and the red dot marking the Zollikofen weather station (USGS, 2023).

2.2 Data

2.2.1 Landsat LST Product

The Landsat LST product is included in the Landsat 8-9 collection 2 level 2 science product, which contains surface reflectance, surface temperature, intermediate bands used for calculating the LST and quality assessment masks. The surface temperatures in this dataset were generated from the Collection 2 Level 1 thermal infrared Sensor (TIRS) band 10, using a single channel algorithm. The calculation includes top-of-atmosphere (TOA) reflectance, TOA brightness temperature and additional data from other sensors, namely the Global emissivity dataset (GED) and the normalised difference vegetation index, both derived from ASTER satellite data. In addition, geopotential height, specific humidity and air temperature from GOES-5 forward processing for instrument teams were included. The surface temperature data are generated at 30m spatial resolution, resampled from 100m resolution of the TIRS sensor on a Universal Transverse Mercator (UTM) mapping grid (USGS, 2022).

The Landsat LST product was validated by Duan et al. (2020) using in situ data. They showed that the Landsat LST product has a mean bias of 1.0K and a root mean square error (RMSE) of 2.1K for snow-free land with correct emissivity estimations. For some surfaces, the emissivity estimations could lead to higher error values.

Landsat data are split up into paths and rows along the satellite's orbit, and the city of Bern is visible on path 195, row 27 and path 196, row 27. The daily acquisition time is 10:16 GMT, which is 11:16 local time in Bern for path 195, row 27 and 10:22 GMT or 11:22 local time for path 196, row 27. All the images for the year 2019 from May to September were downloaded from the "USGS Earthexplorer" (USGS, 2023). All images were checked for cloud contamination visually and with the quality assurance layer included in the product. To transform the data from Band 10 to Kelvin, it was multiplied by 0.00341802 (Multiplicative scale factor), and the additive offset of 149 was added (USGS, 2022). For the year 2019, from May 15 to September 15, four cloud-free images are available, which can be used as input dates and for cross-validation. The dates are the 29 June 2019, 24 July 2019, 9 August 2019, and 25 August 2019. All the images were cut to the desired size (247 x 247 pixels), pixel size at 30m was not changed. The projection of the Landsat images was not changed either. All the pre-processing steps were done using Catalyst and ArcMap (PCI Geomatic Enterprises, 2021; Esri Inc, 2021).

2.2.2 MODIS LST Products

For MODIS, two LST products are available, which use different methods to obtain LST at 1km resolution. Both products are available daily on a global scale and are published by NASA. The two products use different methodologies to calculate LST, and it depends on local properties, which method generates better results.

The MOD11A1 product is based on the generalised split-window algorithm. This approach assigns emissivity based on a land classification scheme. The atmospheric effects are compensated by using two different longwave bands. This approach works exceptionally well in areas where it can be assumed that most of the surface has the same emissivity (Wan, 1999).

The newer MOD21A1 products were released in 2018. The primary difference to MOD11 products is that MOD21 is calculated using the temperature emissivity separation (TES) algorithm. TES uses an emissivity model, which is based on the variability in the surface radiance data (Hulley et al., 2016). It performs better over arid regions than the MOD11, while the performance seems to be very similar in other areas (Hulley et al., 2020).

The MOD11A1 product is used for this study, as many similar studies use MOD11A1 (Liu et al., 2012; Huang et al., 2013; Weng et al., 2014, Li et al., 2016). MOD21A1 only came out much more recently, so it's easier to compare the results with other studies when using the MOD11A1 product. In addition, the MOD21A1 product only performs clearly better in arid regions, which is not the case for the chosen study area. MOD21A1 was only used as a backup if the MOD11A1 had a quality problem or was not available for a particular day.

MODIS data are split up into horizontal and vertical tiles. The city of Bern is located in horizontal tile 18 and vertical tile 4. The acquisition time for these images is at 10:54 GMT or 11:54 local time. All the images used were downloaded on the "USGS Earthexplorer" (USGS, 2023). The images were checked for cloud contamination visually and with the quality assurance layer included in the product. To transform the raw data to Kelvin, all the images were multiplied with a scale factor of 0.02 which can be found in the Collection 6 MODIS Land Surface Temperature Products Users Guide (Wan, 2013). For pre-processing, all the images were resampled to 30m pixels and reprojected to the same UTM mapping grid as the Landsat images. In a second step, they were aligned so that the 30m pixels fit the Landsat images. All pre-processing steps were done in Catalyst and ArcGIS Desktop (PCI Geomatic Enterprises, 2021; Esri Inc, 2021).

2.2.3 Land Cover Data

The land cover data set used to validate the performance of the fusion model is taken from the official cadastral survey of the canton of Bern for the year 2019 (Kanton Bern, 2019). It features land cover and land use data for the canton of Bern. The data are very detailed and include many subcategories that are irrelevant for this analysis. Therefore, the official categories were summarised into five categories of land cover which are of interest for the validation. The summarised categories can be seen in Fig. 2. All land cover data were processed in ArcGIS Desktop (Esri Inc, 2021).

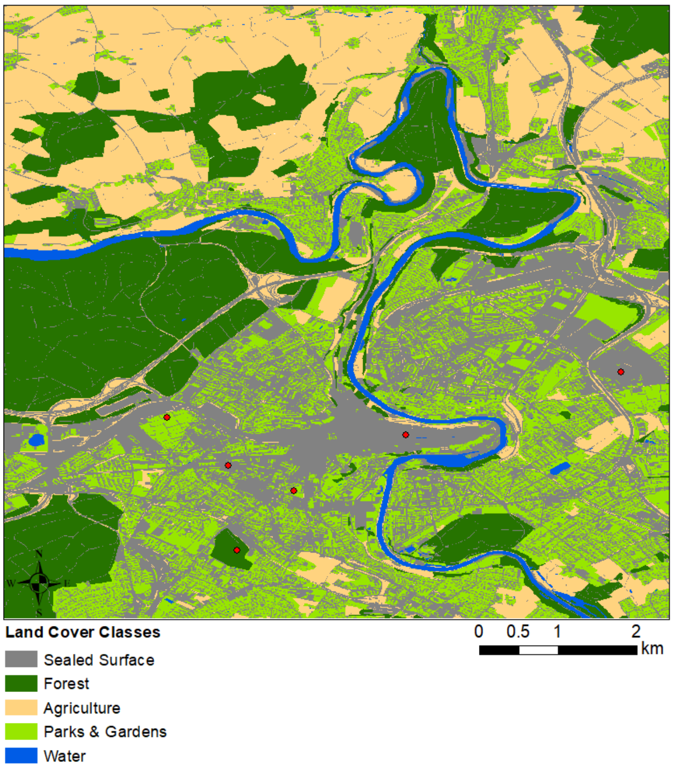


Figure 2 Summarized Landcover classes in the study area with locations of interest for SUHI analysis in red

2.3 Methods

2.3.1 STARFM / ESTARFM Algorithms

This study uses the enhanced STARFM (ESTRAFM) algorithm proposed by Zhu et al. (2010). It includes a conversion coefficient to improve the accuracy in heterogeneous regions, which is a weakness of STARFM (Mao et al., 2021). ESTARFM needs two pairs of high- and low-resolution images at the reference time instead of one and a low-resolution image at the target time. Like STARFM, ESTARFM uses a moving window to find similar pixels and a weight function to predict the value of the centre pixel at the target time (Li *et al.*, 2021). Li et al. (2016) applied ESTARFM in the Fujian Province, China, over a heterogeneous area which includes some urban areas. They used Landsat 8 and MODIS TIR radiance images to perform the fusion and then calculated synthetic Landsat LSTs using a generalised single-channel method introduced by Sobrino et al. (2018). They reported RMSE values between 0.56K and 1.41K compared to Landsat images, depending on the land cover class. However, their study only covers three dates during the winter months. Wu M. et al. (2015) showed that the daily synthetic LSTs generated by ESTARFM had a closer match with actual ASTER observations than STARFM, especially over cultivated land areas. For STARFM, the RMSE values range between 2.35K and 3.38K, while

ESTARFMs RMSE values were slightly better with 1.97K to 3.12K. Compared to the STDFA algorithm proposed by Wu et al. (2012), the performance was very similar. However, over cities and crop areas, ESTARFM worked slightly better. Li et al. (2021) tested, compared and verified STARFM, ESTARFM, and the Flexible Spatiotemporal Data Fusion algorithm by Zhu et al. (2016) in the Gansu Province, China, and directly applied the models to available LST products instead of raw thermal data. They found that all three algorithms showed promising results, and the generated images were consistent when compared with actual Landsat LSTs. Out of the three, ESTARFM showed the highest accuracy in most cases, with an average RMSE of 2.245K compared to real Landsat images.

In Fig. 3, the succession of the ESTARFM algorithm can be seen. In the first step, two fine-resolution Landsat images and two coarse-resolution MODIS images are used to search for pixels similar to the central pixel in a local moving window. The second step calculates the weights of all similar pixels. Third, the conversion coefficients are determined by linear regression. The contribution of similar pixels to predicting reflectance change at the central pixel is determined by the weight. The weight depends on the distance of the similar pixel to the centre pixel and the spectral similarity between the fine- and coarse-resolution pixels. Higher similarity and smaller distance of the similar pixel to the central pixel produce a higher weight. Finally, the conversion coefficient and the weight are used to calculate the fine-resolution reflectance from the coarse-resolution image at the desired prediction date (Zhu *et al.*, 2010).

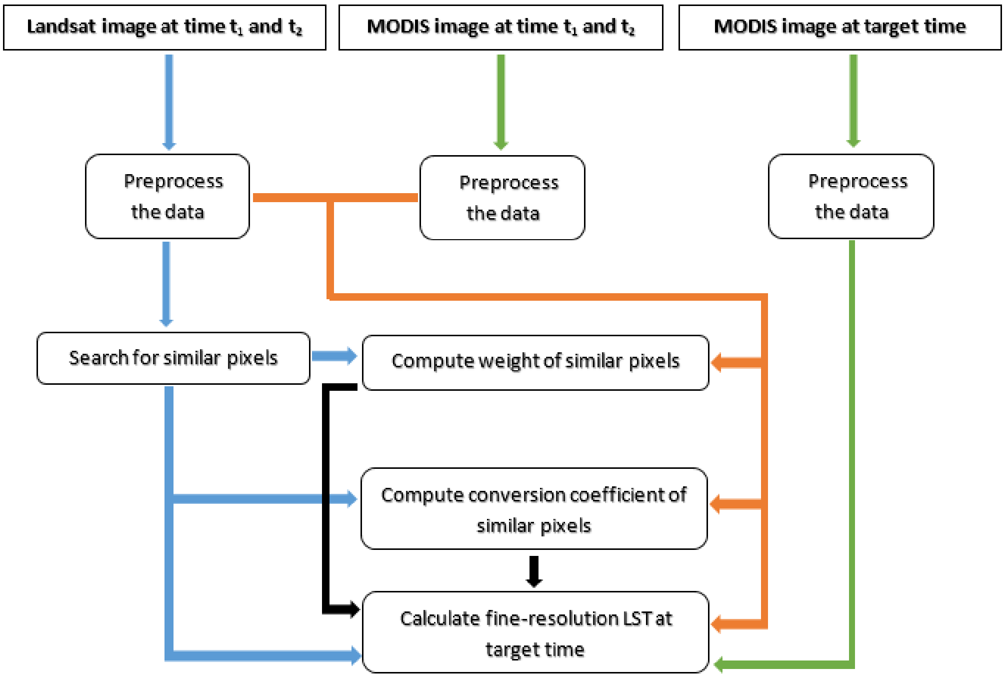

Landsat image at time t_1 and t_2
MODIS image at time t_1 and t_2
MODIS image at target time
Preprocess the data
Preprocess the data
Preprocess the data
Search for similar pixels
Compute weight of similar pixels
Compute conversion coefficient of similar pixels
Calculate fine-resolution LST at target time

Figure 3 Flowchart of the ESTARFM Algorithm succession (Zhu et al.*, 2010)*

Equation (1) shows how ESTARFM predicts a high-resolution Landsat-like LST map from two pairs of high-resolution Landsat images and coarse-resolution MODIS images using a linear spectral mixing model. M represents the high-resolution Landsat images, C the coarse-resolution MODIS images, and w is the size of the moving window. $M(x_{w/2}, y_{w/2}, t_p)$ is the final prediction of high-resolution LST at time t_{p} and $M_k(x_{w/2}, y_{w/2}, t_p)$ is the predicted LST of high spatial resolution at the prediction time t_{p} based on the Landsat 8 LST at time t_k where ($k = m$, n). ($x_{w/2}$, $y_{w/2}$) is the location of the central pixel, (x_i, y_i) is the location of the similar pixel, and N is the number of similar pixels in the searching window. W_i represents the weight of similar pixel *I,* and $V_{i\ is}$ the conversion coefficient of similar pixel *i* (Long *et al.*, 2020).

$$M\left(x_{\frac{w}{2}}, y_{\frac{w}{2}}, t_p\right) = T_m \times M_m\left(x_{\frac{w}{2}}, y_{\frac{w}{2}}, t_p\right) + T_n \times M_n\left(x_{\frac{w}{2}}, y_{\frac{w}{2}}, t_p\right)$$

$$M_k\left(x_{\frac{w}{2}}, y_{\frac{w}{2}}, t_p\right) = M\left(x_{\frac{w}{2}}, y_{\frac{w}{2}}, t_k\right) + \sum_{i=1}^{N} W_i V_i (C(x_i, y_i, t_p) - C(x_i, y_i, t_k))$$

$$(k = m, n)$$

$$(1)$$

Equation (2) shows the definition of the weight W_i, with $\mathrm{D_i}$ representing the variance of the distance vector, which is used for calculating the weights. R*i* is the correlation coefficient between high and low-resolution LST of pixel *i* at t_m and t_n*, with* d_i being the distance weight of pixel *i*. T_k defines the temporal weight (Long *et al.*, 2020).

$$W_i = (1/D_i) \,/ \sum_{i=1}^{N} (1/D_i)$$

$$D_i = (1 - R_i) \times d_i$$

$$R_i = \frac{E[(M_i - E(M_i))(C_i - E(C_i))]}{\sqrt{D(M_i)} \times \sqrt{D(C_i)}}$$

$$d_i = 1 + \sqrt{(x_{\frac{w}{2}} - x_i)^2 + (y_{\frac{w}{2}} - y_i)^2} \,/(w/2)$$

$$T_k = \frac{1/\left|\sum_{j=1}^{W}\sum_{l=1}^{W} C(x_j, y_l, t_k) - \sum_{j=1}^{W}\sum_{l=1}^{W} C(x_j, y_l, t_p)\right|}{\sum_{k=m,n}\left(1/\left|\sum_{j=1}^{W}\sum_{l=1}^{W} C(x_j, y_l, t_k) - \sum_{j=1}^{W}\sum_{l=1}^{W} C(x_j, y_l, t_p)\right|\right)}$$

$$(2)$$

2.3.2 Calibration of ESTARFM

The ESTARFM algorithm was implemented in PyCharm using open-source code based on Zhu et al. (2010). The code is open source and was downloaded from the webpage of (Zhu, 2022). The process needs two Landsat images and two corresponding MODIS images of the same date as the Landsat images and one MODIS image at the desired time when the synthetic Landsat image will be generated.

Two parameters must be set beforehand, based on observation of land cover and experience or trial and error. One is the size of the moving window, and the other is the number of significant land cover classes expected in the study area. The extent of the downscaled area is 247 x 247 pixels large with a spatial resolution of 30m. After a few test runs, a window size of 51 pixels was chosen. The tests showed that with this window size, smaller features like the Aare river are still visible in the downscaled images, but the window is still big enough to have different land cover classes included. The important land cover classes in the study area, with clearly different thermal properties, are built-up areas, forests, agricultural areas and water bodies. Vegetation was differentiated between forest and agricultural areas assuming that agricultural land can change much more during the summer due to crop growth and harvest. At the same time, forested regions are expected to show less change in surface properties. Parks & gardens were separated from forests and agriculture to not introduce error into these categories. However, parks & gardens are not taken into account for the number of significant land cover classes as these areas are mostly very split up and therefore often measured as mixed pixels.

Another critical step is the choice of the reference dates used. Li et al. (2021) generally use the dates closest to the predicted date as, in theory, this should minimise differences in surface cover, solar angle, and sensor difference between predicted and reference images. When using STARFM, where only one reference date is used, they look at the correlation between the MODIS image at prediction time and the two possible adjacent dates to see which one should be used. In this study, for the validation, there are only three reference images available for each validation date. Both methods were tested, and the two adjacent dates show higher correlations compared to the third possible reference date anyway. Therefore, the two adjacent dates will be used when calculating the synthetic images for the target dates where no Landsat image is available.

2.3.3 Validation of ESTARFM

According to Mao et al. (2021), there are three different ways to numerically validate enhanced LST images. These include using simulated data, comparison with ground-based LST measurements and comparison with satellite observations. The ESTARFM method has been validated and discussed in other studies, showing that it works well in different areas (Zhu et al., 2010; Wu M. et al., 2015; Li et al., 2016, 2021). Therefore, in this study, comparisons with actual Landsat LST images will be used for validation, as this is the most straightforward way to see how well the method works under the specific conditions of this particular study area. However, the downside of this validation method is that there are not many possible dates available that can be used for validation. The reason for this is that Landsat has a revisit cycle of 16 days, and cloud contamination reduces the number of suitable images even more. There is no direct way to validate synthetic LSTs for days where no Landsat image is available, which makes sense as this is the reason to perform a downscaling, filling the gaps between actual LST images with predicted ones. It has to be assumed that the validation on the available reference dates is representative for other dates where no Landsat image is available. For the summer of 2019, four Landsat LSTs without cloud contamination are available, which can be used for cross-validation. They can be seen below in Fig. 4, together with the respective MODIS images for the same dates in Fig. 5.

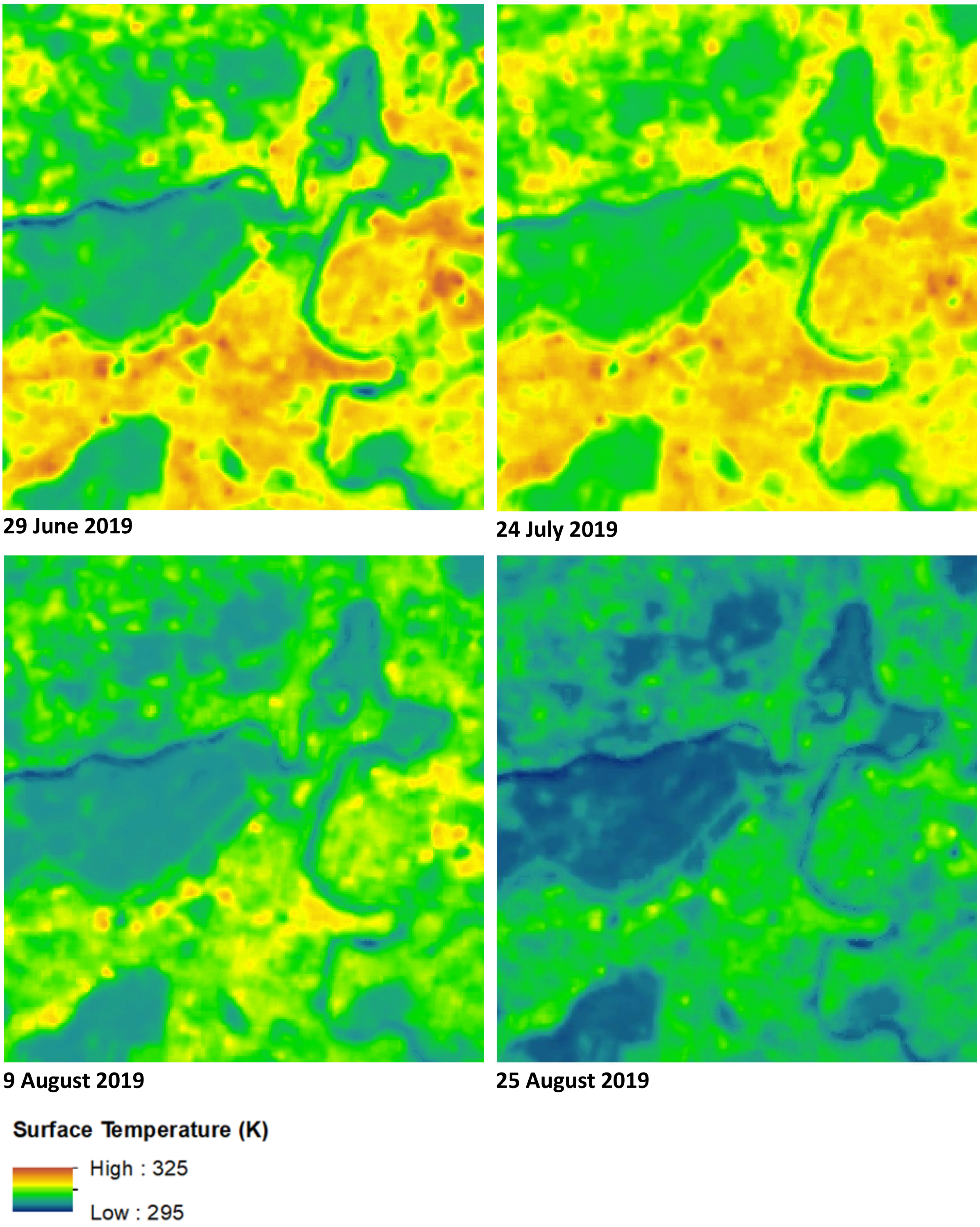


Figure 4 Landsat LST product of the study area for the four reference dates (USGS, 2023).

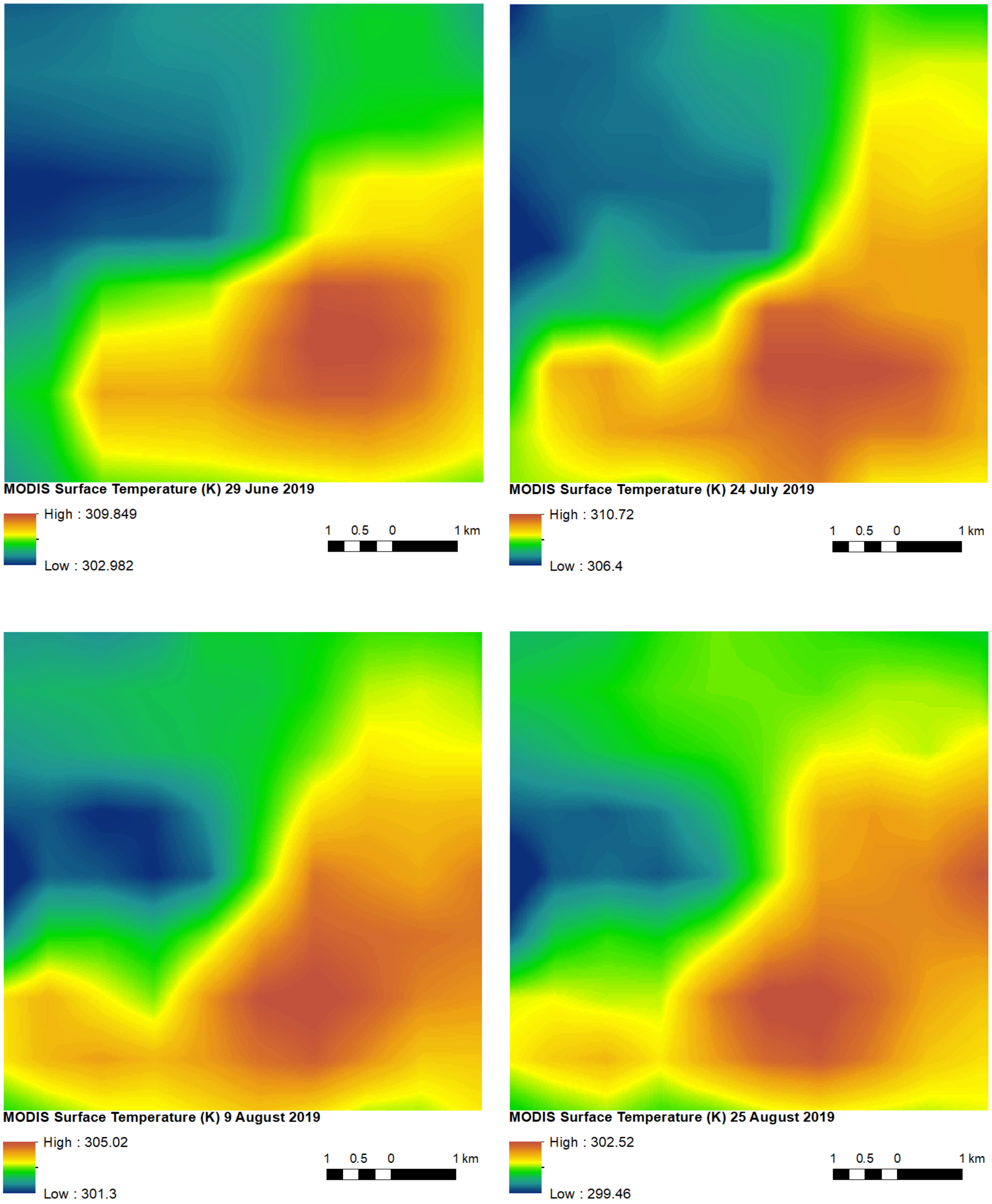

MODIS Surface Temperature (K) 29 June 2019
High : 309.849
Low : 302.982
1 0.5 0 1 km
MODIS Surface Temperature (K) 24 July 2019
High : 310.72
Low : 306.4
1 0.5 0 1 km
MODIS Surface Temperature (K) 9 August 2019
High : 305.02
Low : 301.3
1 0.5 0 1 km
MODIS Surface Temperature (K) 25 August 2019
High : 302.52
Low : 299.46
1 0.5 0 1 km

Figure 5 MODIS LST product of the study area for the four reference dates, resampled to 30m (USGS, 2023).

In the first step, a synthetic Landsat image is calculated for each of the four days with all possible combinations of the other Landsat images as reference dates. This results in having three different predicted Landsat images for each of the four validation dates. By doing this, it is also possible to see the influence of using different reference dates on the downscaling results. Then the generated images can be validated visually by comparing the synthetic and the reference LSTs of the same date. The general structure of the LST distribution and specific features in the study area can be looked at to see if they are preserved in the synthetic images. Especially smaller features can be used to check how well the model performs in heterogeneous areas, as these features are not visible in the MODIS LSTs. See Fig. 5 as an example where the MODIS LST product is shown for the four reference dates. It is not possible to see smaller features like the Aare river in those images, whereas, in the Landsat images with higher spatial resolution, the river can be clearly seen.

In the next step, different numerical and residual related metrics are looked at to compare the predicted and the reference LSTs. The most simple comparisons can be made by just looking at the range, and the mean of the LST values for original and synthetic images Residual related metrics include bias, the mean absolute error (MAE) (Chai and Draxler, 2014), the root mean square error (RMSE) (Chai and Draxler, 2014) and the coefficient of determination (R squared) (Barrett, 2000).

In the last step, the cadastral survey of the canton of Bern (Kanton Bern, 2019) is used to distinguish between different land cover classes in the study area. Like this, the error in built-up areas, forested areas, agricultural areas and water bodies can be compared in isolation to see if the algorithm performs better or worse for specific land cover classes. It is assumed that the land cover, especially in the city, does not change significantly during the summer. Most changes, and therefore the highest errors, probably occur in agricultural areas as crop vegetation changes drastically over the course of a summer. For example, when the crops are harvested, the surfaces change significantly over a very short time.

2.3.4 SUHI Analysis

To analyse the SUHI, synthetic Landsat images are generated over the whole summer of 2019 for each day where a cloud-free MODIS image is available. The generated images are then used to calculate SUHI maps over the entire study area. Depending on the goal of the study, the study area and the scale, the SUHI can be estimated in different ways (Oke *et al.*, 2017). The aim is to look at the structure of the SUHI between different days within a summer. Therefore, a representative point in the rural area is chosen as a reference. The chosen rural point is in Zollikofen, where the official weather station of Bern is located. The station is in an open field surrounded by grassland and crops. To calculate the SUHI, the LST at the reference point is subtracted from all other points in the study area. Using this method, it is possible to directly compare the structure of the SUHI between different days with different conditions. It is important to note that while the LSTs change a lot during the day, this method can only show a snapshot of the SUHI at a specific time when the two satellites measure over Bern, which is between 11:00 and 12:00 local time. Six points within the urban area were chosen to examine the SUHI intensity in more detail. Three are in areas with densely built sealed surfaces where high SUHI intensity is expected. The other three will be in green spaces within the city, like parks or a small forest patch, where lower SUHI intensities are expected. The six points can be seen in Fig. 2. The SUHI intensity can then be compared for the six points between all 47 downscaled images to see how it evolves over the course of the summer.

3 Results

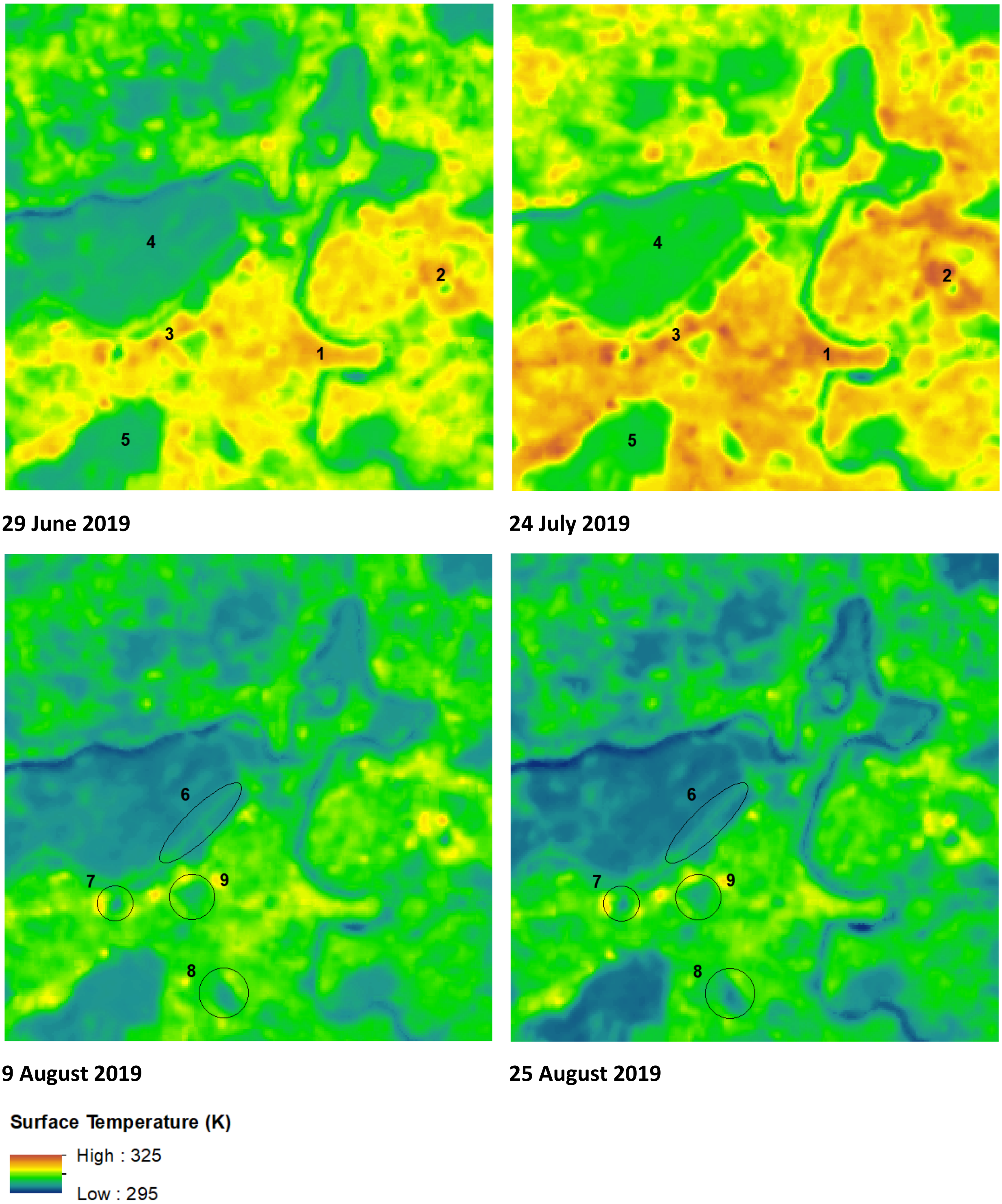
The first part of the results is the validation of the ESTARFM model for the city of Bern to see how the method works in this area, where the most significant errors occur and how much influence the choice of the reference dates has on the results. Land cover data is used to see how the model performs for different surfaces. For this, visual and statistical validation methods will be used. The second part includes the generated time series of synthetic Landsat images for the summer of 2019. An image was generated for each date with a cloud-free, low-resolution MODIS image available. In the last part, the generated high-resolution LST maps will be used to calculate the SUHI intensity for each day. Using these maps, the SUHI can then be analysed, looking at different areas within the city.

3.1 Visual Validation

Fig. 6 shows a synthetic Landsat image for each of the four validation dates, the synthetic images shown as examples are the ones that were calculated using the two adjacent dates as reference. These images are marked in Table 2. with a "*". The residual images in Fig. 7 were calculated by subtracting the synthetic image from the reference Landsat image of the same day. For the residual images, the negative values are pixels where the synthetic Landsat images have higher values than the reference shown in blue, and positive values are pixels where the synthetic Landsat image is colder, shown in red.

In general, the predicted and original Landsat images look very similar. In all the images, the highest temperatures occur in the old town of Bern (1), the Galgenfeld industrial area east of the old town (2) and some scattered areas around Weyermannshaus in the west (3). Bigger forest areas are clearly visible with the coldest surface temperatures in the area. The Bremgartenwald (4) and the Könizbergwald (5) can be seen clearly in every image. Smaller features also seem to be preserved well in the synthetic images. The Aare river and the highway (6) passing the city in the northwest are visible in each image. Other features like the large public swimming pool at Weyermannshaus (7), the Steinhölzliwald (8), and the Bremgartenfriedhof (9) which can all be seen on the original Landsat images are also visible in each downscaled image. The heat patterns in the old town, where some of the highest surface temperatures are measured, look very similar between the different dates. In contrast, the patterns in the rural parts north and south-east of the city show more differences.

The residuals images show that the highest errors occur in the rural area north of the city (10) / (11) for all dates. The land cover classes in Fig. 2 show that these are predominantly agricultural areas with villages scattered in between. Forest areas show the lowest errors and the least variation for most dates. Built-up areas show similar spatial error patterns in all images. However, they can be overestimated (24 July and 25 August) or underestimated (29 June and 9 August). Each image also has a relatively large error in the Viererfeld (12) north of the old town. Viererfeld is an isolated patch of agricultural area in which the surface changes significantly during summer. The Aare river also shows positive and negative error values depending on the date. In the residual images of the 9 August and 25 August, the area southeast of the city (13) shows that in some images, very similar patterns can occur but with opposite signs. Another pattern that is visible in all the residual images is the oblique grid structure all over the images.



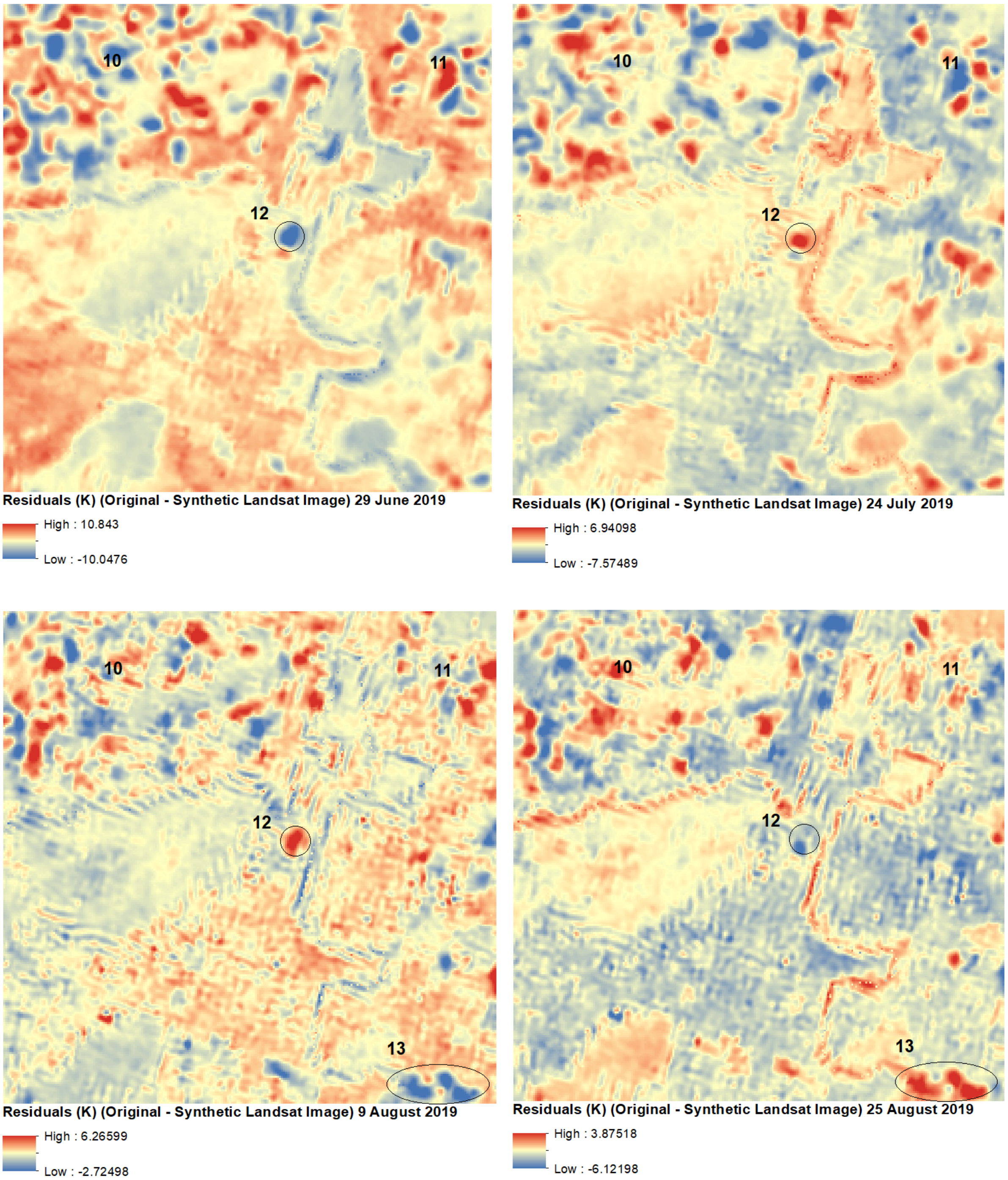
29 June 2019

24 July 2019

9 August 2019

25 August 2019

Figure 6 Synthetic Landsat images generated by ESTARFM for the four reference dates.



Residuals (K) (Original - Synthetic Landsat Image) 29 June 2019

Residuals (K) (Original - Synthetic Landsat Image) 24 July 2019

Residuals (K) (Original - Synthetic Landsat Image) 9 August 2019

Residuals (K) (Original - Synthetic Landsat Image) 25 August 2019

Figure 7 Residual images (Original – Synthetic) for the four reference dates.

3.2 Statistical Validation

Table 1 shows the mean values of the original and synthetic Landsat images. For the synthetic images, the table shows the runs where adjacent dates were used as reference, so the images which can be seen in Fig 6. The difference in mean values is mostly around 1K. More importantly, it clearly shows that the order of temperatures is preserved; the hottest day of the original images is also the hottest for the synthetic images. When comparing the temperature ranges of the synthetic images and the original Landsat images (Table 2), a similar picture can be seen, but there are some differences. For the original Landsat images, the 29 June and the 24 July show almost the same maximum temperature but in the synthetic images, the maximum on the 29 June is underestimated, while on the 24 July, it is overestimated. Interestingly, the minimum values for the same dates show the opposite.

Table 1 Mean values of the original and synthetic images for the four validation dates.

Date	Mean original Landsat (K)	Mean synthetic Landsat (K)
29.6.2019	309.808	308.827
24.7.2019	311.160	312.091
9.8.2019	306.375	305.098
25.8.2019	302.666	304.161

Table 2 Min and Max values for the original and synthetic images for the four validation dates.

Date	Min original Landsat (K)	Min synthetic Landsat (K)	Max original Landsat (K)	Max synthetic Landsat (K)
29.6.2019	296.908	297.945	323.719	321.396
24.7.2019	301.051	299.071	323.343	324.417
9.8.2019	296.792	296.656	317.608	315.751
25.8.2019	295.011	294.801	313.027	314.878

Table 3 shows the RMSE, MAE, R squared and bias for each validation date. It includes each day three times using all the possible combinations of reference dates as input. The images with the "*" are the ones where the input reference dates are closest to the target date, which is the same configuration that will be used for the time series.

The predicted images for the 24 July and the 9 August yield the best results for every combination of reference dates when looking at RMSE and MAE, while the 29 June and the 25 August show higher values for all combinations of reference dates. The best prediction is the 24 July (9 August / 25 August) with an RMSE of 1.043K and MAE of 0.763K; the worst prediction is the 25 August (29 June / 24 July) with RMSE and MAE values reaching 2.471K and 2.099K. On average, over all 12 validation images, the RMSE reaches 1.811K and the MAE 1.476K. For the 29 June, the bias shows that for all combinations, the predicted images show smaller values than the actual Landsat image. In contrast, for the 25 August, the prediction seems too high compared to the reference Landsat image. For the 24 July and the 9 August, there is both positive and negative bias depending on the reference dates used.

The R-squared values are very similar for all combinations on the 29 June and the 24 July, while the other two dates vary much more depending on the chosen reference days.

Table 3 RMSE, MAE, R Squared and Bias for the four reference dates of the synthetic LSTS with all combinations of possible reference dates.

Date and reference dates	**RMSE(K)**	**MAE(K)**	**R Squared**	**Bias(K)**
29.6 (24.7 / 9.8)*	2.108	1.642	0.866	-0.981
29.6 (24.7 / 25.8)	2.446	1.978	0.866	-1.582
29.6 (9.8 / 25.8)	2.400	1.907	0.841	-1.280
24.7 (29.6 / 9.8)*	1.557	1.247	0.920	0.932
24.7 (29.6 / 25.8)	1.475	1.117	0.905	0.584
24.7 (9.8 / 25.8)	1.043	0.763	0.929	-0.194
9.8 (29.6 / 24.7)	1.634	1.199	0.855	0.333
9.8 (29.6 / 25.8)	1.292	0.999	0.901	-0.532
9.8 (24.7 / 25.8)*	1.472	1.318	0.964	-1.276
25.8 (29.6 / 24.7)	2.471	2.099	0.855	1.855
25.8 (29.6 / 9.8)	2.114	1.897	0.935	1.836
25.8 (24.7 / 9.8)*	1.721	1.552	0.949	1.495
Mean	**1.811**	**1.477**	**0.899**	**0.099**

** = Adjacent dates*

3.3 Land Cover Classes

To analyse the errors specifically for the different land cover classes, the best and the worst prediction image for each validation date were taken by looking at the RMSE and MAE values in Table 3. The summarised land cover classes from the cadastral survey are forest, agriculture, parks & gardens, sealed surface and water (Fig. 2). Vegetation classes were separated into three categories as it is assumed that agriculture areas change much during the summer due to crop growth and harvest. In contrast, forested areas do not change much. Parks & gardens were separated from the other vegetation classes to get a clearer picture of the forested and agricultural areas because they do not fit the other two categories. Table 4 shows each Land cover class's mean RMSE and Bias over all twelve images of the four reference dates. The full tables with separate values for the best and worst prediction for each reference date can be found in the attachments section I. Forest shows the lowest error overall, with RMSE values ranging from 0.581K to 1.427K, while agricultural areas show the highest RMSE values ranging from 1.559K to 3.093K. Parks & gardens and sealed surfaces are in between, with average RMSE values of 1.680K and 1.834K, respectively. Bias values show positive and negative signs for all land cover classes on individual images. On average, however, with the exception of water, they all are slightly positive.

Table 4 RMSE and Bias of the land cover classes for the reference dates.

Land Cover Class	Mean RMSE (K)	Mean Bias (K)
Forest	0.997	0.108
Agriculture	2.334	0.170
Parks & Gardens	1.680	0.187
Sealed Surface	1.834	0.211
Water	1.257	-0.005

3.4 SUHI time series

After the cross-validation of the reference dates, a time series was calculated for every day between 15 March and 15 September 2019, where a cloud-free MODIS image was available. Out of the 123 days, 47 of the corresponding MODIS Images were cloud-free and could be used to generate a synthetic Landsat image. Using the synthetic Landsat images, the SUHI was calculated by subtracting the value of the rural reference point “Zollikofen Referenz”. Fig 8. shows two examples; the 30 June is the day with the highest SUHI intensity, reaching a maximum value of 16.4K compared to the rural reference. The 4 August has the lowest SUHI intensity with a maximum SUHI value of approximately 11K. There are also pixels with much lower surface temperatures than “Zollikofen Referenz”, mostly in forested areas. The minimal values seem to be independent of the strength of the SUHI in the city, as for both images shown, the minimum values are -10.908K and -10.544K respectively.

When just looking at the temperature patterns and comparing them with the landcover classes in Fig. 2, the higher SUHI values seem to correlate strongly with sealed surfaces in the city. Densely built areas like the old town and industrial areas show clear hotspots. The surface urban heat island is not one big, connected area. Many smaller, mostly vegetated areas with lower SUHI intensities are scattered between warmer areas. Also, the Aare river has a significant impact, splitting the SUHI into two parts. As mentioned above, the synthetic images preserved the smaller structures within the city quite well. This can also be seen in the SUHI maps, showing lower values within the city in areas with more vegetation, for example the Bremgartenfriedhof or the Steinhölzliwald (Fig. 8).

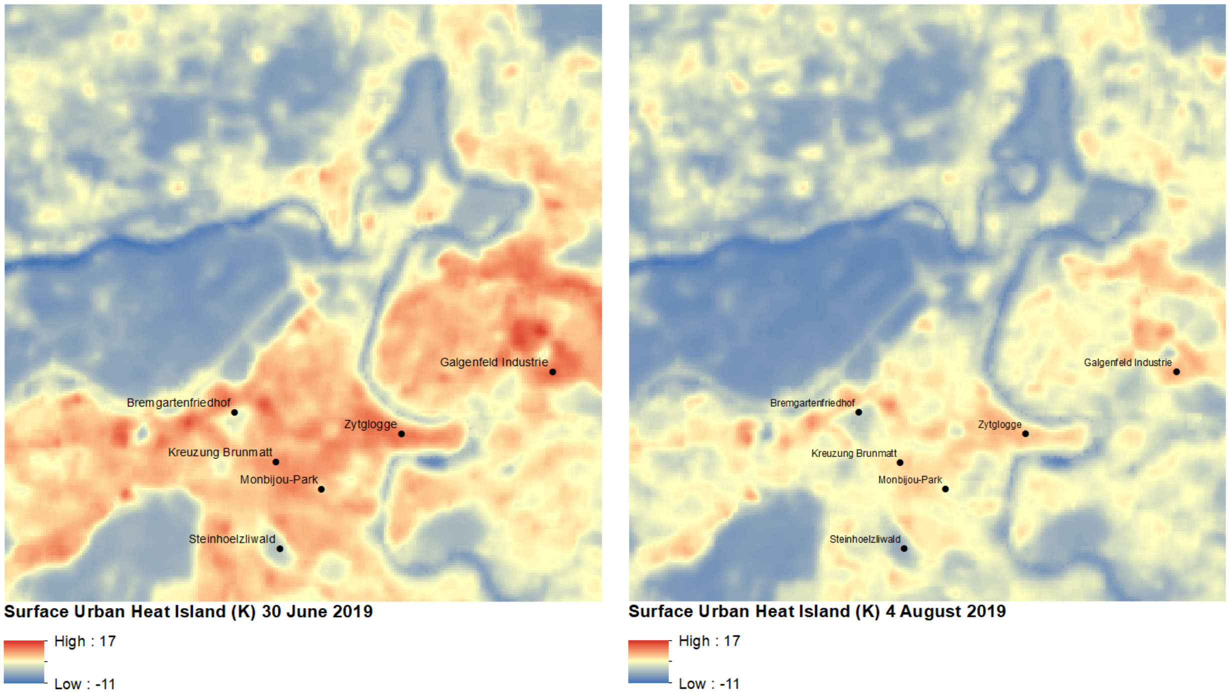


Figure 8 Surface urban heat island map from synthetic LSTs for the 30 June 2019 and the 4 August 2019.

In the next step, six points were selected to look at the SUHI values within the city in more detail. The points can be seen in Fig 8. Three of them are hotspots, the Zytglogge in the old town, Kreuzung Brunnmatt west of the old town and Galgenfeld Industrie in the industrial area northeast of the city centre. The other three points are spots within the urban area with more vegetation. One is the Steinhölzliwald, a small forest area in the southern part of the city, which is surrounded by sealed surfaces. The other two points are the Bremgartenfriedhof, a rather large graveyard park with many trees and green areas and the Monbijoupark, which is a smaller park compared to the Bremgartenfriedhof but still visible in the synthetic images.

The four reference dates used before were taken again to calculate the SUHI for the original and synthetic Landsat images. The resulting SUHI maps were then subtracted from each other to compare how much they differ for each day. Fig. 9 shows the difference in SUHI intensities when calculated with the original Landsat images and synthetic ones for the six chosen points in the city. The differences in SUHI intensity between the original and synthetic Landsat images range from -2.259K to 1.710K, with an absolute average of 0.778K over all six points on the four reference dates. Here the same thing can be seen as in the statistical validation. The 24 July and 9 August seem to have much better predictions than the other dates.

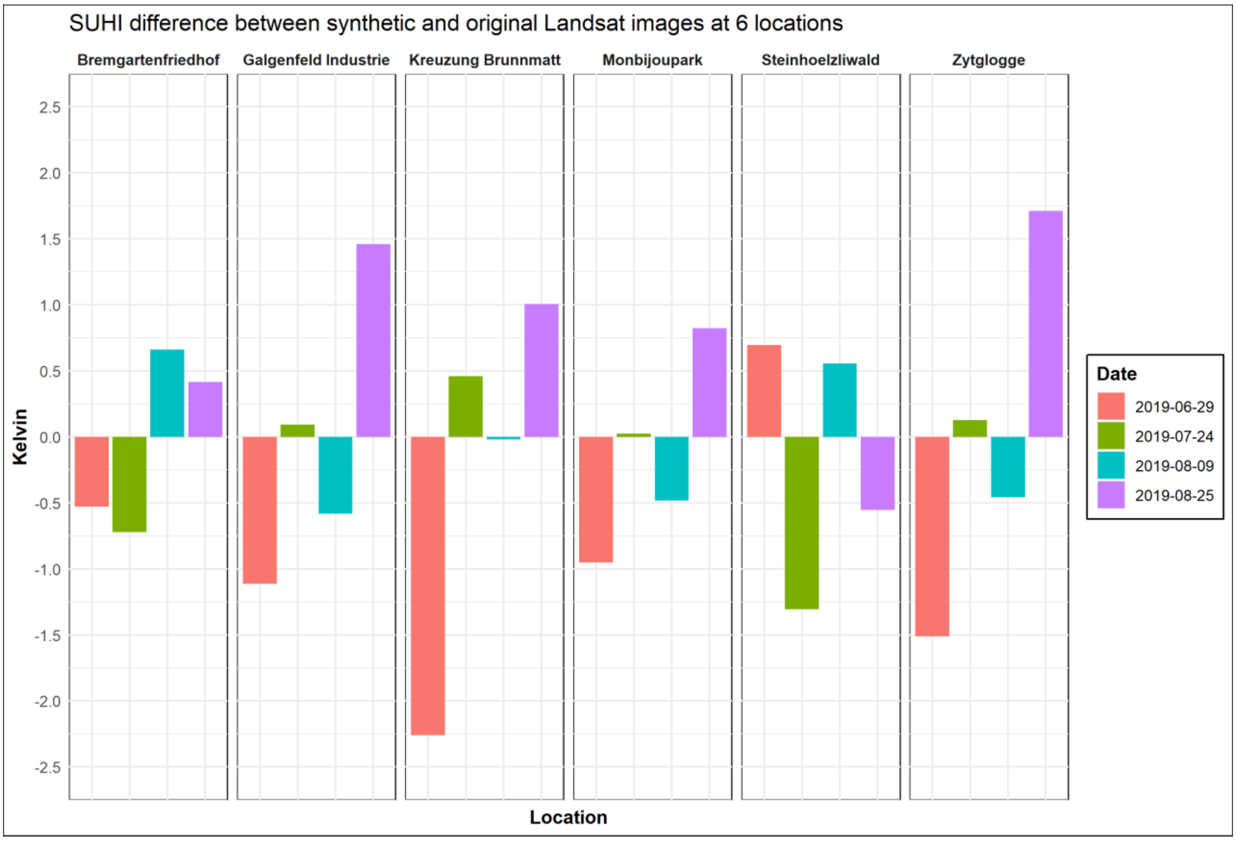


Figure 9 Difference of the SUHI when comparing synthetic and original Landsat images for the four reference dates at six chosen points within the city.

In the last step, a time series was generated for the whole summer of 2019 at each of the six chosen points within the city. Fig. 10 shows the time series for the six points on all 47 cloud-free days. The three points in built up areas show higher SUHI values than the other three points every day. Except for the Steinhölzliwald, all locations show a positive SUHI effect, but the two parks within the city show clearly lower values compared to the points in built-up areas. The three hotspots have SUHI values between 5.002K and 12.594K during the summer; the two parks range between -0.136K and 6.272K, and in the Steinhölzliwald, the values are between -4.721K and 0.080K. A downward trend is visible for the five points showing a SUHI effect, starting in July until the end of the summer. At the end of August, there is a strong peak, where 2-3 days show much higher SUHI values than adjacent dates. In general, the values show high similarities; even the station Steinhölzliwald shows similar variations to the other stations, but the downward trend over the course of the summer is not visible.

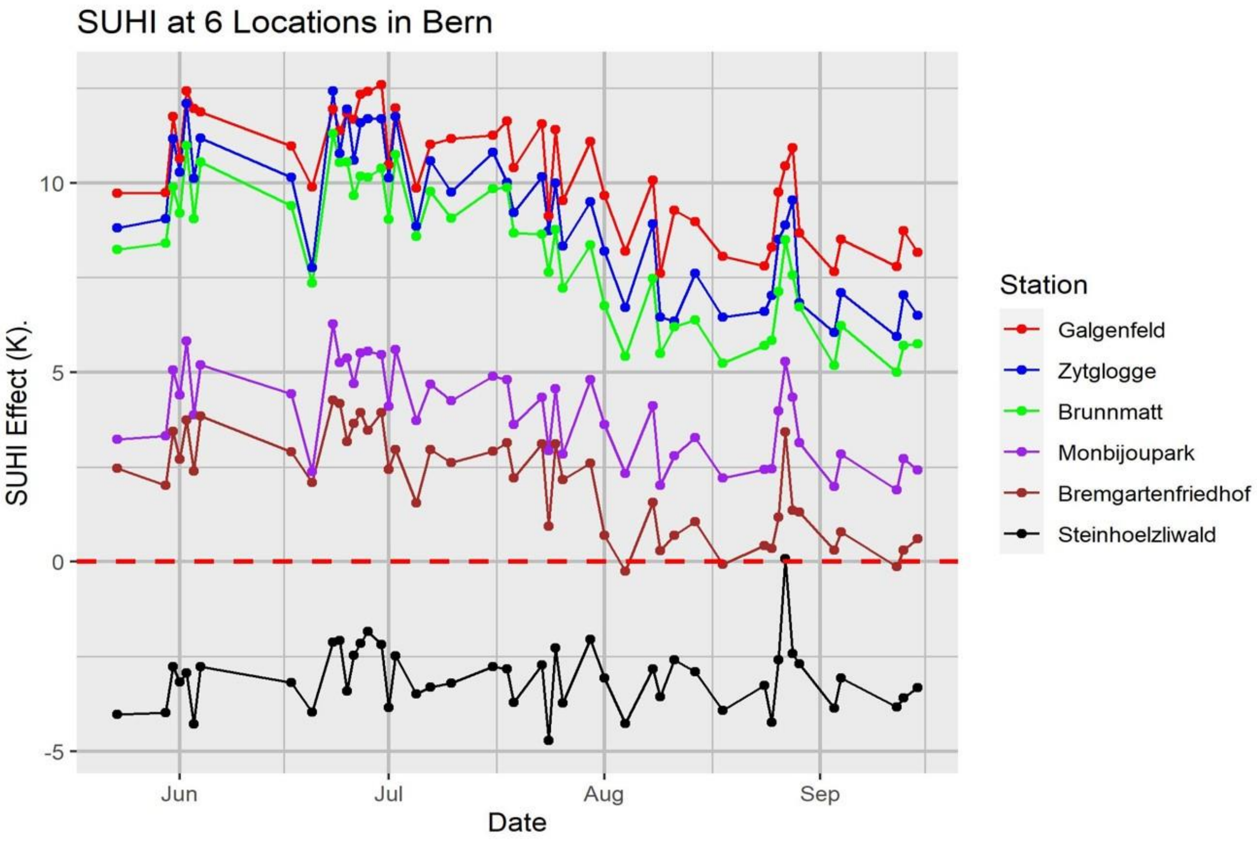


Figure 10 Time series of the SUHI effect at six selected points in the summer of 2019.

4 Discussion

In the first part, the accuracy of the generated images is discussed by looking at possible sources of error and the algorithm's performance in different land cover classes. The second part discusses the suitability of the generated images for SUHI analysis in a city like Bern and looks at some limitations and possible improvements of ESTARFM and fusion-based methods in general.

4.1 Accuracy of the Generated Images

In general, the fused images look very similar to the original ones. Small features that cannot be seen in the MODIS images are visible after the downscaling, for example, the Aare river or different parks in the city. A clear pattern is visible in all the synthetic images, showing that the city's densely built, central parts heat up the most compared to other areas. This can be seen very well when comparing the sealed surface areas in Fig. 2 with the generated images in Fig. 6. Looking at the statistical metrics, the focus will be on the RMSE as this metric is used in most other studies and is therefore most suited for comparisons. With RMSE values ranging from 1.043K to 2.471K and an average of 1.811K for the twelve validated images, ESTARFM performed better in Bern and its surroundings than in cultivated land areas in Zhangye, China, where the study of Wu et al. (2015) showed RMSE values between 1.97K and 3.12K. However, while their study also looked at the summer months, it must be noted that the study area was bigger, and they used ASTER images instead of Landsat. Li et al. (2021) also used ESTARFM in Zhangye, China, over a smaller area. They used Landsat and MODIS images and reported average RMSE values of 2.245K, which is very close to the results of this study. Other than the study area itself, the main difference here is that they looked at images between April and October. Li et al. (2016) applied ESTARFM with Landsat and MODIS images in the Fujian Province, China showing RMSE values between 0.56K and 1.41K. Their study is set in the winter months, and there are barely any agricultural surfaces in the study area which could explain the better results. Huang et al. (2013) presented a study over urban area in Beijing where they used a method based on STARFM where a bilateral filter is introduced to account for the effect of neighbouring pixels in urban areas. While the process differs slightly from ESTARFM, the study used Landsat and MODIS images and yielded similar results to this study with an average RMSE of 1.699K.

This study finds the best matches in forested areas and the highest errors in agricultural areas. This is expected because forested areas should change the least during the summer, as phenological changes in trees happen primarily in spring and autumn. In addition to that, forested areas are protected in Switzerland, and while they are used and cultivated to a certain extent, there are rarely large deforestations over short times. A third factor is that forested areas heat up slowly, so the time between the overpasses of the two different satellites should generate less error compared to other areas. Agricultural areas on the other hand, change much more between May and September due to the crop's phenological changes and the harvest. The crops grow over time and cover a bigger part of the soil. After the crops are harvested, bare soil dominates again. These changes are difficult to catch when only having four Landsat scenes as reference, which can lead to some errors in the predictions. This fits the findings of Li et al. (2021), who also reported larger errors in areas where there is land cover change between the reference and the predicted images when using ESTARFM. When looking at sealed surface areas, higher errors can be found compared to forested areas. A possible explanation other than changes in surface cover could be that sealed surfaces have a much larger temperature amplitude during a day and are much more sensitive to fast changes. Sealed surfaces heat up quickly when exposed to direct radiation, while forested areas take up the heat much slower due to different albedo, evapotranspiration, heat capacity and conductivity (Stewart, 2011). This can introduce some errors for sealed surfaces as the two satellites don’t pass precisely at the same time, and even a difference of just 30 minutes to 1 hour can be enough to heat sealed surfaces for several Kelvin when exposed to direct radiation. While sealed surfaces themselves should not change a lot, urban areas are much more sensitive to different observation angles of the satellites and different solar angles during a summer due to the complicated 3D structures of cities (Oke *et al.*, 2017). The categories parks & gardens and water are more difficult to interpret. These two categories are much more scattered than the others and only get measured as mixed pixels on MODIS images. Therefore, their temperatures are assumed to be much more dependent on the Landsat reference images as they are mostly seen as mixed pixels, especially in the MODIS images. Another source of error is the choice of reference images. As shown in the validation, for the same day, there are RMSE differences between 0.341K and 0.750K, just depending on which reference images were chosen for a specific date. In theory, it should be best to choose images closest to the target date (Zhu *et al.*, 2010). However, the validation of the reference dates shows that this not always yields the best result. A possible reason for this could be the different ranges of temperatures in the reference dates, resulting from differing meteorological conditions each day. For example, rainfall or the number of sunshine hours on the previous days could impact LSTs.

Another influence that must be discussed is the changing solar angles over the year. For example, at the end of August, the solar angle is smaller than in July. Therefore, the surfaces should heat up slower during the morning which can be relevant considering that the satellites overpasses are between 11:00 and 12:00 local time. This trend of a slower heating with smaller solar angles can be seen in Fig. 10, starting in mid-July. While all the images are cloud-free, there are still hotter and cooler days depending on the abovementioned factors. If the difference between the two references is large, this could also generate an error even in areas with little change in the surface itself. This would also explain why there are positive and negative signs of bias for all different surfaces. A possible explanation could be that large temperature differences between the two reference images lead to favouring one of the two high-resolution reference images when calculating the weights. This could lead to similar patterns in the synthetic images of different dates where at least one of the used reference images was the same. An example would be the residual images of the 9 August / 25 August in Fig. 7, where almost identical error patterns can be seen in some parts of the image. Another pattern that can be seen in the residual images are the oblique grid structures over all the images. They are most likely an artefact from resampling the 1000m Modis images to 30m.

In Fig.10, there are some very distinctive peaks visible. The question is whether those peaks represent different meteorological conditions for these days or if they result from an error caused by large temperature differences between the two reference days. Because there are no original Landsat images for these days to verify, the weather station data of two stations in Bern were checked. The station data for the days at the end of August, where a strong peak is visible, confirms that these days were exceptionally hot, with a high amount of sunshine hours and no precipitation. For the peak in mid-June, the air temperatures at midday also show lower values, and there is also some precipitation in the previous days which could slow down the heating of the surface due to increased latent heat fluxes caused by evapotranspiration. This confirms that the synthetic images are sensitive to the meteorological conditions despite them only being measured by the coarse resolution MODIS images for the prediction day.

4.2 Suitability and Limitations of ESTARFM for SUHI Analysis

The generated images are similar enough to original Landsat images to be useful for SUHI analysis, even for a small city like Bern. Important features are preserved, and even small parks surrounded by sealed surfaces are still visible in the generated images. The maps can be helpful in identifying hotspots of LST in a city. Looking at the land cover data, the strong impact of sealed surfaces on the LSTs can be seen clearly. The influence of green areas within the urban area can also be visualised well using these maps. The difference between the calculated SUHI values of original and synthetic Landsat images seems to be pretty low, considering the large range the surface temperatures can show over a small area compared to air temperatures. Looking at Fig. 10, the SUHI values that were calculated using the generated images seem reasonable, with values between 5.002K and 12.594K. In comparison, Parlow et al. (2014) reported SUHI values between 5K and 10K in Basel, Switzerland, where they used Landsat thermal data. In their review of SUHI studies, Zhou et al. (2018) found that the maximum SUHI intensity in summer during the day in most studies is larger than 10K, which also fits the results found in Bern using the generated LST maps. Another positive of the method is that some errors are predictable quite well, for example in areas where it is known that they undergo changes in surface cover over a short time. This was seen especially well in agricultural areas.

There are also some important limitations to this method of resolution enhancement. First, it can only be used for images that are not cloud contaminated. While there are some methods to exclude cloud pixels for scattered clouds, it is not possible to gain information about LST on full overcast days. This is not a problem when analysing SUHI since the intensity and the effect of the SUHI are not as relevant on days with many clouds. In future studies where overcast days are of interest, reconstruction techniques for cloudy pixels could be applied, as discussed by Wu et al. (2021). Another problem is the error propagation during a fusion-based downscaling. It is important to remember that the validation done here is against the original Landsat image, which is not an absolute ground truth. Landsat and MODIS LST products also have different errors and uncertainties, and they get propagated when doing an image fusion. Therefore, in a more detailed study analysing the SUHI, it would be necessary to compare the LST products and the image fusion results with in situ data measured on the ground or by a drone.

Looking at urban areas specifically, the complicated 3D structures of an urban surface can also not be forgotten. Due to these complex 3D structures of buildings, vegetation and other features, it is very difficult for a satellite-based sensor to see the entire surface. The resulting LSTs are only a result of what the sensors can measure, so primarily the horizontal surfaces, while the vertical surfaces get neglected. According to Oke et al. (2017), this can make a big difference when calculating SUHI intensity. If vertical surfaces not registered by the satellite would be included, the SUHI could decrease up to 3K for daytime measurements. To correct this, much higher spatial resolutions would be needed to generate detailed 3D maps of the city. This could only be possible using drones or other aircraft able to measure with higher spatial resolutions. Another possible step in future studies to improve the quality of LST maps could be to use the generated maps as input for a kernel-based downscaling. Such methods have been gaining popularity where fusion-based and kernel-based methods are combined to further increase spatial and temporal resolution (Xia *et al.*, 2019).

5 Conclusion

In this study, the Landsat and MODIS LST products were combined using ESTARFM to generate Landsat-like images of surface temperature for the city of Bern in the summer of 2019. The generated images were validated by comparing them with actual Landsat images of the same days. The synthetic Landsat images look very similar to real Landsat images. They are in the range of other studies using ESTARFM with an average RMSE value of 1.811K for the twelve compared images from four different days. In the second part, a synthetic Landsat image was generated for 47 cloud-free days during the summer of 2019. The 47 images were then used to look at the structure and development of the SUHI in the urban area of Bern. The SUHI was calculated by subtracting a chosen rural reference station from the whole image. Six different locations in the city were selected to compare the SUHI values. Three of them are in densely built-up areas like the old town, in an industrial area and in the middle of the city. As expected, these areas have continuously higher SUHI values compared to the rural station, showing a SUHI effect of 5K-12K over the summer. In the Park areas, the surface temperatures are still higher than at the rural station, with values between 0K and 6K. In contrast, the SUHI effect is not present in the forest patch, and the measured surface temperatures are continuously lower compared to the rural reference. To conclude, this study shows that the ESTARFM method is suited to generate synthetic Landsat images for a small city like Bern to gain more information about surface temperatures. While there are still some limitations to the method, the results are satisfying and can be used to calculate the SUHI for days without Landsat overflights. Possible improvements could be made by validating the input images of Landsat and MODIS for the study area, specifically by using in situ data. Another improvement in future studies could be using data measured by drones with higher resolution to gain more information about the 3D structure of the city and the combination of the fusion-based images with a kernel-based method to further improve the spatial resolution. To increase the temporal resolution, cloud pixel reconstruction methods could also be applied.

Acknowledgements

First, I would like to thank my supervisors Prof. Dr. Stefan Brönnimann, Prof. Dr. Stefan Wunderle, and Dr. Moritz Gubler, who gave me the possibility to write this thesis in their research groups. Second, I want to thank Moritz, Sonia, Nils, Christoph and everyone else on the 5th floor for their inputs on different methodological, GIS and programming-related questions. Last, I want to thank my family, my dog and especially Sabine for their constant support during my thesis.

References

Barrett, G.B. (2000) ‘The Coefficient of Determination: Understanding r squared and R squared’, *The Mathematics Teacher*, 93(3), pp. 230–234. Available at: https://doi.org/10.5951/MT.93.3.0230.

Burger, M.A. *et al.* (2021) *Modelling the spatial pattern of the urban heat island of Bern during heatwaves in 2018 and 2019 using a land use regression approach Urban Climate*, 38(100885). Available at: https://doi.org/10.1016/j.uclim.2021.100885

Chai, T. and Draxler, R.R. (2014) ‘Root mean square error (RMSE) or mean absolute error (MAE)?’, *Geosci. Model Dev. Discuss*, 7, pp. 1525–1534. Available at: https://doi.org/10.5194/gmdd-7-1525-2014.

Duan, S.B. *et al.* (2020) ‘Validation of Landsat land surface temperature product in the conterminous United States using in situ measurements from SURFRAD, ARM, and NDBC sites’, *International Journal of Digital Earth*, 14(5), pp. 640–660. Available at: https://doi.org/10.1080/17538947.2020.1862319.

EEA (European Environment Agency) (2022) ‘Climate change as a threat to health and well-being in Europe: focus on heat and infectious diseases’. Available at: https://doi.org/10.2800/67519.

Esri Inc (2021) ‘ArcGis Desktop 10.8.2’.

Gao, F. *et al.* (2006) ‘On the blending of the landsat and MODIS surface reflectance: Predicting daily landsat surface reflectance’, *IEEE Transactions on Geoscience and Remote Sensing*, 44(8), pp. 2207–2218. Available at: https://doi.org/10.1109/TGRS.2006.872081.

Gubler, M. *et al.* (2021) ‘Evaluation and application of a low-cost measurement network to study intra-urban temperature differences during summer 2018 in Bern, Switzerland’, *Urban Climate*, 37(100817). Available at: https://doi.org/10.1016/J.UCLIM.2021.100817.

Huang, B. *et al.* (2013) ‘Generating high spatiotemporal resolution land surface temperature for urban heat island monitoring’, *IEEE Geoscience and Remote Sensing Letters*, 10(5), pp. 1011–1015. Available at: https://doi.org/10.1109/LGRS.2012.2227930.

Hulley, G., Malakar, N. and Freepartner, R. (2016) ‘Moderate Resolution Imaging Spectroradiometer (MODIS) Land Surface Temperature and Emissivity Product (MxD21) Algorithm Theoretical Basis Document Collection-6’, pp. 12–17.

Hulley, G. and Nickeson, J. (2020) ‘M*D21 (MYD21/MOD21) and M*D11 (MYD11/MYD21) Land Surface Temperature and Emissivity (LST&E) Products Status Report’.

IPCC 2022 (2022a) *IPCC 2022: Cities, Settlements and Key Infrastructure. In: Climate Change 2022: Impacts, Adaptation, and Vulnerability.*, *Climate Change 2022: Impacts, Adaptation and Vulnerability. Contribution of Working Group II to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change [H.-O. Pörtner, D.C. Roberts, M. Tignor, E.S. Poloczanska, K. Mintenbeck, A. Alegr*. Available at: https://doi.org/10.1017/9781009325844.008.

IPCC 2022 (2022b) *IPCC 2022, Summary for Policymakers*, *Climate Change 2022: Impacts, Adaptation and Vulnerability. Contribution of Working Group II to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change H.-O. Pörtner, D.C. Roberts, M. Tignor, E.S. Poloczanska, K. Mintenbeck, A. Alegrí*. Cambridge University Press, Cambridge, UK and New York. Available at: https://doi.org/10.1017/9781009325844.001.

Kanton Bern (2019) *Amtliche Vermessung*. Available at: https://www.agi.dij.be.ch/de/start/kataster/amtliche-vermessung/daten-der-amtlichen-vermessung.html (Accessed: 22 February 2023).

Li, Q. *et al.* (2016) 'Improvement of ESTARFM and Its Application to Fusion of Landsat-8 and MODIS Land Surface Temperature Images', in *4th International Workshop on Earth Observation and Remote Sensing Applications, EORSA 2016 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., pp. 33–37. Available at: https://doi.org/10.1109/EORSA.2016.7552761.

Li, S. *et al.* (2021) 'Evaluation of landsat 8-like land surface temperature by fusing landsat 8 and modis land surface temperature product', *Processes*, 9(2262). Available at: https://doi.org/10.3390/pr9122262.

Liu, H. and Weng, Q. (2012) 'Enhancing temporal resolution of satellite imagery for public health studies: A case study of West Nile Virus outbreak in Los Angeles in 2007', *Remote Sensing of Environment*, 117, pp. 57–71. Available at: https://doi.org/10.1016/j.rse.2011.06.023.

Long, D. *et al.* (2020) 'Generation of MODIS-like land surface temperatures under all-weather conditions based on a data fusion approach', *Remote Sensing of Environment*, 246(111863). Available at: https://doi.org/10.1016/J.RSE.2020.111863.

Mao, Q., Peng, J. and Wang, Y. (2021) 'Resolution enhancement of remotely sensed land surface temperature: Current status and perspectives', *Remote Sensing*, 13(1306). Available at: https://doi.org/10.3390/rs13071306.

MeteoSchweiz (2021) *Klimawandel - MeteoSchweiz*. Available at: https://www.meteoschweiz.admin.ch/klima/klimawandel.html (Accessed: 6 December 2022).

MeteoSchweiz (2022a) *Meteo Schweiz Climate Overview 1991-2020 Bern*. Available at: https://www.meteoschweiz.admin.ch/service-und-publikationen/applikationen/ext/climate-climsheet.html (Accessed: 20 December 2022).

MeteoSchweiz (2022b) *Wärmstes und sonnigstes Jahr seit Messbeginn*. Available at: https://www.meteoschweiz.admin.ch/ueber-uns/meteoschweiz-blog/de/2022/12/waermstes-und-sonnigstes-jahr-seit-messbeginn.html (Accessed: 10 March 2023).

Oke, T.R. *et al.* (2017) 'Urban climates - 7. Urban Heat Island', in *Urban Climates*. Cambridge University Press, pp. 197–237. Available at: https://doi.org/10.1017/9781139016476.

Parlow, E. (2013) 'Urban Climate', in *Urban Ecology*. Oxford University Press, pp. 31–44. Available at: https://doi.org/10.1093/acprof:oso/9780199563562.003.0005.

Parlow, E., Vogt, R. and Feigenwinter, C. (2014) 'The urban heat island of Basel - Seen from different perspectives', *Erde*, 145(1–2), pp. 96–110. Available at: https://doi.org/10.12854/ERDE-145-8.

PCI Geomatic Enterprises (2021) 'Catalyst Professional Version 2222.0.0'.

Sobrino, J.A. *et al.* (2018) 'An Improved Single-Channel Method to Retrieve Land Surface Temperature from the Landsat-8 Thermal Band', *Remote Sensing 2018, Vol. 10, Page 431*, 10(3), p. 431. Available at: https://doi.org/10.3390/RS10030431.

Stewart, I.D. (2011) *Redefining the Urban Heat Island*, University of British Columbia. Available at https://doi.org/10.14288/1.0072360

USGS (2022) *Landsat 8-9 Collection 2 (C2) Level 2 Science Product (L2SP) Guide (Version 4.0)*.

USGS (2023) *Earth Explorer*. Available at: https://earthexplorer.usgs.gov/ (Accessed: 10 March 2023).

Wan, Z. (1999) *MODIS Land-Surface Temperature Algorithm Theoretical Basis Document (LST ATBD) (Version 3.3)*. Santa Barbara.

Wan, Z. (2013) 'Collection-6 MODIS Land Surface Temperature Products Users' Guide', *ERI, University of California, Santa Barbara*.

Weng, Q., Fu, P. and Gao, F. (2014) 'Generating daily land surface temperature at Landsat resolution by fusing Landsat and MODIS data', *Remote Sensing of Environment*, 145, pp. 55–67. Available at: https://doi.org/10.1016/j.rse.2014.02.003.

Wu, M. *et al.* (2012) 'Use of MODIS and Landsat time series data to generate high-resolution temporal synthetic Landsat data using a spatial and temporal reflectance fusion model', *Journal of Applied Remote Sensing*, 6(063507–1). Available at: https://doi.org/10.1117/1.JRS.6.063507.

Wu, M. *et al.* (2015) 'Generating daily high spatial land surface temperatures by combining ASTER and MODIS land surface temperature products for environmental process monitoring', *Environmental Science: Processes & Impacts*, 17(8), pp. 1396–1404. Available at: https://doi.org/10.1039/C5EM00254K.

Wu, P. *et al.* (2015) 'Integrated fusion of multi-scale polar-orbiting and geostationary satellite observations for the mapping of high spatial and temporal resolution land surface temperature', *Remote Sensing of Environment*, 156, pp. 169–181. Available at: https://doi.org/10.1016/j.rse.2014.09.013.

Wu, P. *et al.* (2021) 'Spatially Continuous and High-Resolution Land Surface Temperature Product Generation: A review of reconstruction and spatiotemporal fusion techniques', *IEEE Geoscience and Remote Sensing Magazine*, 9(3), pp. 112–137. Available at: https://doi.org/10.1109/MGRS.2021.3050782.

Xia, H. *et al.* (2019) 'Combining kernel-driven and fusion-based methods to generate daily high-spatial-resolution land surface temperatures', *Remote Sensing of Environment*, 224, pp. 259–274. Available at: https://doi.org/10.1016/J.RSE.2019.02.006.

Yao, R. *et al.* (2018) 'The influence of different data and method on estimating the surface urban heat island intensity', *Ecological Indicators*, 89, pp. 45–55. Available at: https://doi.org/10.1016/J.ECOLIND.2018.01.044.

Zhou, D. *et al.* (2018) 'Satellite Remote Sensing of Surface Urban Heat Islands: Progress, Challenges, and Perspectives', *Remote Sensing 2019, Vol. 11, Page 48*, 11(1), p. 48. Available at: https://doi.org/10.3390/RS11010048.

Zhu, X. *et al.* (2010) 'An enhanced spatial and temporal adaptive reflectance fusion model for complex heterogeneous regions', *Remote Sensing of Environment*, 114(11), pp. 2610–2623. Available at: https://doi.org/10.1016/j.rse.2010.05.032.

Zhu, X. *et al.* (2016) 'A flexible spatiotemporal method for fusing satellite images with different resolutions', *Remote Sensing of Environment*, 172, pp. 165–177. Available at: https://doi.org/10.1016/J.RSE.2015.11.016.

Zhu, X. (2022) *Open-Source Code - Polyu remote sensing intelligence for dynamic Earth (Pride)*. Available at: https://xiaolinzhu.weebly.com/open-source-code.html (Accessed: 20 July 2022).

Attachments

I – Land cover validation full table

Forest

Date and Reference	RMSE(K)	Bias(K)
29.6 (24.7 / 9.8)	0.999	-0.210
29.6 (24.7 / 25.8)	1.280	-0.733
24.7 (29.6 / 9.8)	0.743	0.426
24.7 (9.8 / 25.8)	0.581	-0.040
9.8 (29.6 / 24.7)	0.910	-0.100
9.8 (29.6 / 25.8)	0.855	-0.593
25.8 (29.6 / 24.7)	1.427	1.036
25.8 (24.7 / 9.8)	1.182	1.081
Mean	**0.997**	**0.108**

Agriculture

Date and Reference	RMSE(K)	Bias(K)
29.6 (24.7 / 9.8)	2.949	-0.716
29.6 (24.7 / 25.8)	3.093	-1.491
24.7 (29.6 / 9.8)	2.066	0.817
24.7 (9.8 / 25.8)	1.559	-0.142
9.8 (29.6 / 24.7)	2.449	0.259
9.8 (29.6 / 25.8)	1.933	-0.475
25.8 (29.6 / 24.7)	2.756	1.694
25.8 (24.7 / 9.8)	1.864	1.414
Mean	**2.334**	**0.170**

Parks & Gardens

Date and Reference	RMSE(K)	Bias(K)
29.6 (24.7 / 9.8)	1.860	-1.375
29.6 (24.7 / 25.8)	2.387	-1.950
24.7 (29.6 / 9.8)	1.471	1.134
24.7 (9.8 / 25.8)	0.931	-0.293
9.8 (29.6 / 24.7)	1.316	0.525
9.8 (29.6 / 25.8)	1.060	-0.545
25.8 (29.6 / 24.7)	2.580	2.270
25.8 (24.7 / 9.8)	1.832	1.729
Mean	**1.680**	**0.187**

Sealed surface

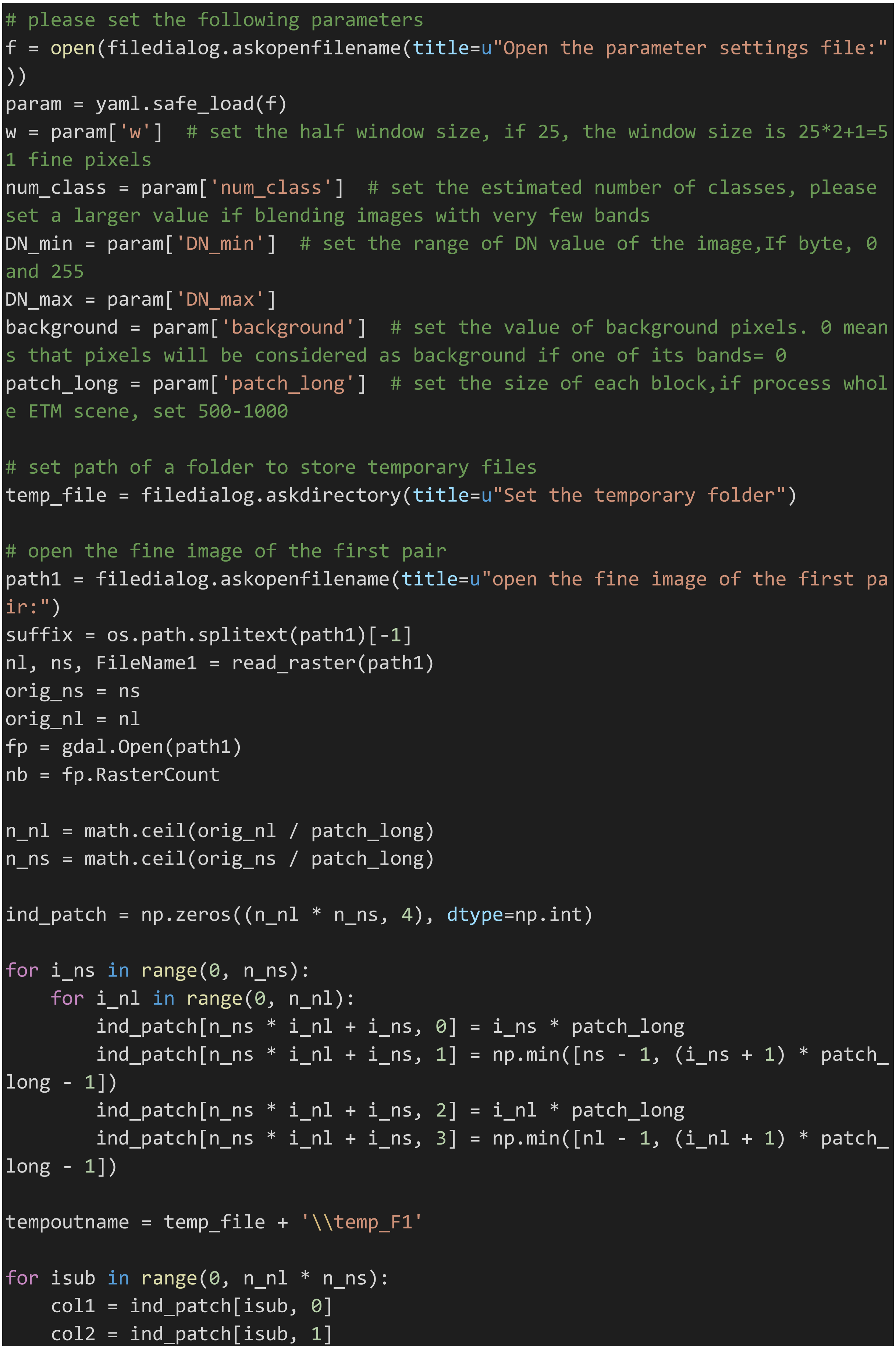
Date and Reference	RMSE(K)	Bias(K)
29.6 (24.7 / 9.8)	2.155	-1.721
29.6 (24.7 / 25.8)	2.675	-2.282
24.7 (29.6 / 9.8)	1.658	1.395
24.7 (9.8 / 25.8)	0.869	-0.363
9.8 (29.6 / 24.7)	1.448	0.740
9.8 (29.6 / 25.8)	1.063	-0.492
25.8 (29.6 / 24.7)	2.869	2.576
25.8 (24.7 / 9.8)	1.934	1.833
Mean	**1.834**	**0.211**

Water

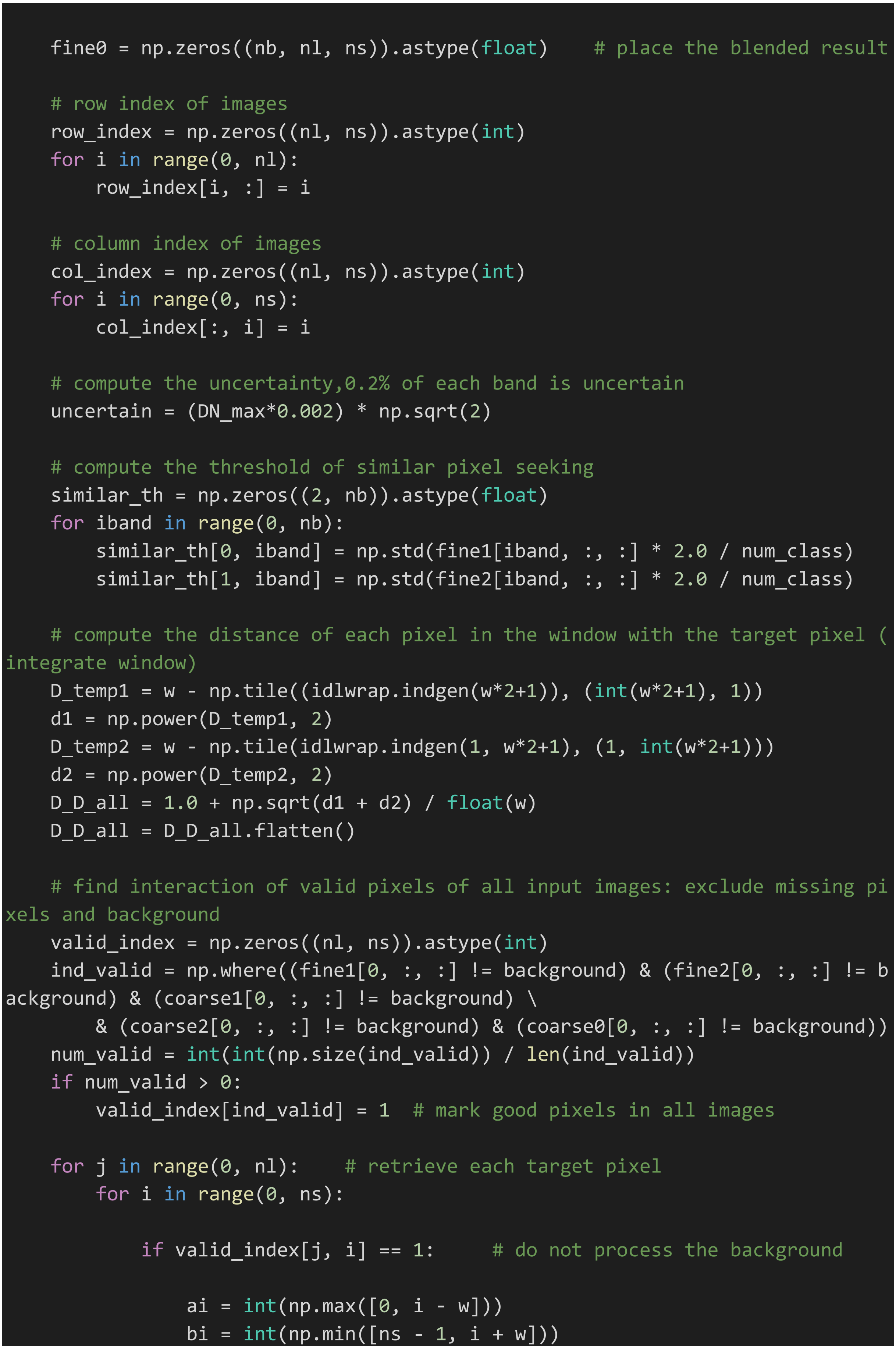
Date and Reference	**RMSE(K)**	**Bias(K)**
29.6 (24.7 / 9.8)	1.518	0.724
29.6 (24.7 / 25.8)	1.467	0.377
24.7 (29.6 / 9.8)	1.034	-0.028
24.7 (9.8 / 25.8)	0.973	0.552
9.8 (29.6 / 24.7)	1.555	-1.00
9.8 (29.6 / 25.8)	1.183	-0.839
25.8 (29.6 / 24.7)	1.384	-0.322
25.8 (24.7 / 9.8)	0.940	0.498
Mean	**1.257**	**-0.005**

***II** – Full python code ESTARFM*

```
from utils import read_raster, writeimage
import math
import numpy as np
from osgeo import gdal
import os
import datetime
from tkinter import filedialog
import tkinter as tk
import yaml
import idlwrap
import statsmodels.api as sm
from scipy.stats import f
from sklearn.linear_model import LinearRegression

# ****************************************************************************
**************************
#                            ESTARFM PROGRAM
#               Using two pairs of fine and coarse images
#         the program can be used for whole TM scene and VI index product

# ****************************************************************************
**************************
# *******************************Set parameters and read input data***********
**************************

root = tk.Tk()
root.withdraw()
```



```python
# please set the following parameters
f = open(filedialog.askopenfilename(title=u"Open the parameter settings file:"))
param = yaml.safe_load(f)
w = param['w']  # set the half window size, if 25, the window size is 25*2+1=51 fine pixels
num_class = param['num_class']  # set the estimated number of classes, please set a larger value if blending images with very few bands
DN_min = param['DN_min']  # set the range of DN value of the image,If byte, 0 and 255
DN_max = param['DN_max']
background = param['background']  # set the value of background pixels. 0 means that pixels will be considered as background if one of its bands= 0
patch_long = param['patch_long']  # set the size of each block,if process whole ETM scene, set 500-1000

# set path of a folder to store temporary files
temp_file = filedialog.askdirectory(title=u"Set the temporary folder")

# open the fine image of the first pair
path1 = filedialog.askopenfilename(title=u"open the fine image of the first pair:")
suffix = os.path.splitext(path1)[-1]
nl, ns, FileName1 = read_raster(path1)
orig_ns = ns
orig_nl = nl
fp = gdal.Open(path1)
nb = fp.RasterCount

n_nl = math.ceil(orig_nl / patch_long)
n_ns = math.ceil(orig_ns / patch_long)

ind_patch = np.zeros((n_nl * n_ns, 4), dtype=np.int)

for i_ns in range(0, n_ns):
    for i_nl in range(0, n_nl):
        ind_patch[n_ns * i_nl + i_ns, 0] = i_ns * patch_long
        ind_patch[n_ns * i_nl + i_ns, 1] = np.min([ns - 1, (i_ns + 1) * patch_long - 1])
        ind_patch[n_ns * i_nl + i_ns, 2] = i_nl * patch_long
        ind_patch[n_ns * i_nl + i_ns, 3] = np.min([nl - 1, (i_nl + 1) * patch_long - 1])

tempoutname = temp_file + '\\temp_F1'

for isub in range(0, n_nl * n_ns):
    col1 = ind_patch[isub, 0]
    col2 = ind_patch[isub, 1]
    row1 = ind_patch[isub, 2]
    row2 = ind_patch[isub, 3]
    data = FileName1[:, row1:row2 + 1, col1:col2 + 1]
    out_name = tempoutname + str(isub + 1) + suffix
    fp = path1
    writeimage(data, out_name, fp)

# open the coarse image of the first pair
path2 = filedialog.askopenfilename(title=u"open the coarse image of the first
pair:")
_, _, FileName2 = read_raster(path2)

tempoutname = temp_file + '\\temp_C1'
for isub in range(0, n_nl * n_ns):
    col1 = ind_patch[isub, 0]
    col2 = ind_patch[isub, 1]
    row1 = ind_patch[isub, 2]
    row2 = ind_patch[isub, 3]
    data = FileName2[:, row1:row2 + 1, col1:col2 + 1]
    out_name = tempoutname + str(isub + 1) + suffix
    fp = path1
    writeimage(data, out_name, fp)

# open the fine image of the second pair
path3 = filedialog.askopenfilename(title=u"open the fine image of the second p
air:")
_, _, FileName3 = read_raster(path3)

tempoutname = temp_file + '\\temp_F2'
for isub in range(0, n_nl * n_ns):
    col1 = ind_patch[isub, 0]
    col2 = ind_patch[isub, 1]
    row1 = ind_patch[isub, 2]
    row2 = ind_patch[isub, 3]
    data = FileName3[:, row1:row2 + 1, col1:col2 + 1]
    out_name = tempoutname + str(isub + 1) + suffix
    fp = path1
    writeimage(data, out_name, fp)

# open the coarse image of the second pair
path4 = filedialog.askopenfilename(title=u"open the coarse image of the second
 pair:")
_, _, FileName4 = read_raster(path4)

tempoutname = temp_file + '\\temp_C2'
for isub in range(0, n_nl * n_ns):
    col1 = ind_patch[isub, 0]
    col2 = ind_patch[isub, 1]
    row1 = ind_patch[isub, 2]
    row2 = ind_patch[isub, 3]
    data = FileName4[:, row1:row2 + 1, col1:col2 + 1]
    out_name = tempoutname + str(isub + 1) + suffix
    fp = path1
    writeimage(data, out_name, fp)

# open the coarse image of the prediction time
path5 = filedialog.askopenfilename(title=u"open the coarse image of the predic
tion time:")
_, _, FileName5 = read_raster(path5)

tempoutname = temp_file + '\\temp_C0'
for isub in range(0, n_nl * n_ns):
    col1 = ind_patch[isub, 0]
    col2 = ind_patch[isub, 1]
    row1 = ind_patch[isub, 2]
    row2 = ind_patch[isub, 3]
    data = FileName5[:, row1:row2 + 1, col1:col2 + 1]
    out_name = tempoutname + str(isub + 1) + suffix
    fp = path1
    writeimage(data, out_name, fp)

# *******************************************************
# process each clock
# *******************************************************

starttime = datetime.datetime.now()  # the initial time of program running
print('there are total', n_nl*n_ns, 'blocks')

for isub in range(0, n_nl * n_ns):

    # open each block image

    FileName = temp_file + '\\temp_F1' + str(isub + 1) + suffix
    nl, ns, fine1 = read_raster(FileName)

    FileName = temp_file + '\\temp_C1' + str(isub + 1) + suffix
    _, _, coarse1 = read_raster(FileName)

    FileName = temp_file + '\\temp_F2' + str(isub + 1) + suffix
    _, _, fine2 = read_raster(FileName)

    FileName = temp_file + '\\temp_C2' + str(isub + 1) + suffix
    _, _, coarse2 = read_raster(FileName)

    FileName = temp_file + '\\temp_C0' + str(isub + 1) + suffix
    _, _, coarse0 = read_raster(FileName)
```



```
    fine0 = np.zeros((nb, nl, ns)).astype(float)    # place the blended result

    # row index of images
    row_index = np.zeros((nl, ns)).astype(int)
    for i in range(0, nl):
        row_index[i, :] = i

    # column index of images
    col_index = np.zeros((nl, ns)).astype(int)
    for i in range(0, ns):
        col_index[:, i] = i

    # compute the uncertainty,0.2% of each band is uncertain
    uncertain = (DN_max*0.002) * np.sqrt(2)

    # compute the threshold of similar pixel seeking
    similar_th = np.zeros((2, nb)).astype(float)
    for iband in range(0, nb):
        similar_th[0, iband] = np.std(fine1[iband, :, :] * 2.0 / num_class)
        similar_th[1, iband] = np.std(fine2[iband, :, :] * 2.0 / num_class)

    # compute the distance of each pixel in the window with the target pixel (integrate window)
    D_temp1 = w - np.tile((idlwrap.indgen(w*2+1)), (int(w*2+1), 1))
    d1 = np.power(D_temp1, 2)
    D_temp2 = w - np.tile(idlwrap.indgen(1, w*2+1), (1, int(w*2+1)))
    d2 = np.power(D_temp2, 2)
    D_D_all = 1.0 + np.sqrt(d1 + d2) / float(w)
    D_D_all = D_D_all.flatten()

    # find interaction of valid pixels of all input images: exclude missing pixels and background
    valid_index = np.zeros((nl, ns)).astype(int)
    ind_valid = np.where((fine1[0, :, :] != background) & (fine2[0, :, :] != background) & (coarse1[0, :, :] != background) \
        & (coarse2[0, :, :] != background) & (coarse0[0, :, :] != background))
    num_valid = int(int(np.size(ind_valid)) / len(ind_valid))
    if num_valid > 0:
        valid_index[ind_valid] = 1  # mark good pixels in all images

    for j in range(0, nl):    # retrieve each target pixel
        for i in range(0, ns):

            if valid_index[j, i] == 1:     # do not process the background

                ai = int(np.max([0, i - w]))
                bi = int(np.min([ns - 1, i + w]))
                aj = int(np.max([0, j - w]))
                bj = int(np.min([nl - 1, j + w]))

                ind_wind_valid = np.where((valid_index[aj:bj+1, ai:bi+1]).rave
l() == 1)
                position_cand = idlwrap.intarr((bi-ai+1)*(bj-
aj+1)) + 1    # place the location of each similar pixel
                row_wind = row_index[aj:bj+1, ai:bi+1]
                col_wind = col_index[aj:bj + 1, ai:bi + 1]

                # searching for similar pixels
                for ipair in [0, 1]:
                    for iband in range(0, nb):
                        cand_band = idlwrap.intarr((bi-ai+1)*(bj-aj+1))
                        if ipair == 0:
                            S_S = np.abs(fine1[iband, aj:bj+1, ai:bi+1] - fine
1[iband, j, i])
                        elif ipair == 1:
                            S_S = np.abs(fine2[iband, aj:bj + 1, ai:bi + 1] -
fine2[iband, j, i])
                        ind_cand = np.where(S_S.ravel() < similar_th[ipair, ib
and])
                        cand_band[ind_cand] = 1
                        position_cand = position_cand * cand_band

                cand_band = 0
                indcand = np.where((position_cand != 0) & ((valid_index[aj:bj+
1, ai:bi+1]).ravel() == 1))
                number_cand = int(int(np.size(indcand)) / len(indcand))

                if number_cand > 5:     # compute the correlation
                    S_D_cand = np.zeros(number_cand).astype(float)
                    x_cand = (col_wind.ravel())[indcand]
                    y_cand = (row_wind.ravel())[indcand]
                    finecand = np.zeros((nb*2, number_cand)).astype(float)
                    coarsecand = np.zeros((nb*2, number_cand)).astype(float)

                    for ib in range(0, nb):
                        finecand[ib, :] = (fine1[ib, aj:bj+1, ai:bi+1]).ravel(
)[indcand]
                        finecand[ib+nb, :] = (fine2[ib, aj:bj+1, ai:bi+1]).rav
el()[indcand]
                        coarsecand[ib, :] = (coarse1[ib, aj:bj+1, ai:bi+1]).ra
vel()[indcand]
                        coarsecand[ib+nb, :] = (coarse2[ib, aj:bj+1, ai:bi+1])
.ravel()[indcand]

                    if nb == 1:    # for images with one band, like NDVI
```

```
                        S_D_cand = 1.0 - 0.5*(np.abs((finecand[0, :]-
coarsecand[0, :]) / (finecand[0, :]+coarsecand[0, :])) +
                                              np.abs((finecand[1, :]-
coarsecand[1, :]) / (finecand[1, :]+coarsecand[1, :])))
                    else:
                        # for images with multiple bands
                        sdx = np.std(finecand, axis=0, ddof=1)
                        sdy = np.std(coarsecand, axis=0, ddof=1)
                        meanx = np.mean(finecand, axis=0)
                        meany = np.mean(coarsecand, axis=0)

                        x_meanx = np.zeros((nb*2, number_cand)).astype(float)
                        y_meany = np.zeros((nb*2, number_cand)).astype(float)
                        for ib in range(0, nb*2):
                            x_meanx[ib, :] = finecand[ib, :] - meanx
                            y_meany[ib, :] = coarsecand[ib, :] - meany

                        S_D_cand = nb*2.0*np.mean(x_meanx*y_meany, axis=0) / (
sdx*sdy) / (nb*2.0-1)

                    ind_nan = np.where(S_D_cand != S_D_cand)
                    num_nan = int(int(np.size(ind_nan)) / len(ind_nan))
                    if num_nan > 0:
                        S_D_cand[ind_nan] = 0.5    # correct the NaN value of
correlation

                    D_D_cand = np.zeros(number_cand).astype(float)   # spatial
 distance
                    if (bi-ai+1)*(bj-
aj+1) < (w*2.0+1)*(w*2.0+1):   # not an integrate window
                        D_D_cand = 1.0 + np.sqrt((i-x_cand)**2+(j-
y_cand)**2) / w
                    else:
                        D_D_cand[0:number_cand] = D_D_all[indcand]      # inte
grate window

                    C_D = (1.0-
S_D_cand) * D_D_cand + 0.0000001           # combined distance
                    weight = (1.0/C_D)/np.sum(1.0/C_D)

                    for ib in range(0, nb):   # compute V
                        fine_cand = np.hstack(((fine1[ib, aj:bj+1, ai:bi+1]).r
avel()[indcand], (fine2[ib, aj:bj+1, ai:bi+1]).ravel()[indcand]))
                        coarse_cand = np.hstack(((coarse1[ib, aj:bj+1, ai:bi+1
]).ravel()[indcand], (coarse2[ib, aj:bj+1, ai:bi+1]).ravel()[indcand]))
                        coarse_change = np.abs(np.mean((coarse1[ib, aj:bj+1, a
i:bi+1]).ravel()[indcand]) - np.mean((coarse2[ib, aj:bj+1, ai:bi+1]).ravel()[i
ndcand]))
```

```
                        if coarse_change >= DN_max*0.02:  # to ensure changes in coarse image large enough to obtain the conversion coefficient

                            X = coarse_cand.reshape(-1, 1)
                            Y = fine_cand.reshape(-1, 1)
                            XX = sm.add_constant(X)
                            model = sm.OLS(Y, XX).fit()
                            regress_result = model.params
                            sig = model.f_pvalue

                            # correct the result with no significancy or inconsistent change or too large value
                            if sig <= 0.05 and 0 < regress_result[1] <= 5:
                                V_cand = regress_result[1]
                            else:
                                V_cand = 1.0

                        else:
                            V_cand = 1.0

                        # compute the temporal weight
                        difc_pair1 = np.abs(np.mean((coarse0[ib, aj:bj+1, ai:bi+1]).ravel()[ind_wind_valid])-np.mean((coarse1[ib, aj:bj+1, ai:bi+1]).ravel()[ind_wind_valid]))+0.01**5
                        difc_pair2 = np.abs(np.mean((coarse0[ib, aj:bj+1, ai:bi+1]).ravel()[ind_wind_valid])-np.mean((coarse2[ib, aj:bj+1, ai:bi+1]).ravel()[ind_wind_valid]))+0.01**5
                        T_weight1 = (1.0/difc_pair1) / (1.0/difc_pair1+1.0/difc_pair2)
                        T_weight2 = (1.0/difc_pair2) / (1.0/difc_pair1+1.0/difc_pair2)

                        # predict from pair1
                        coase0_cand = (coarse0[ib, aj:bj+1, ai:bi+1]).ravel()[indcand]
                        coase1_cand = (coarse1[ib, aj:bj+1, ai:bi+1]).ravel()[indcand]
                        fine01 = fine1[ib, j, i] + np.sum(weight * V_cand * (coase0_cand-coase1_cand))
                        # predict from pair2
                        coase2_cand = (coarse2[ib, aj:bj+1, ai:bi+1]).ravel()[indcand]
                        fine02 = fine2[ib, j, i] + np.sum(weight * V_cand * (coase0_cand-coase2_cand))
                        # the final prediction
                        fine0[ib, j, i] = T_weight1 * fine01 + T_weight2 * fine02
                        # revise the abnormal prediction
                        if fine0[ib, j, i] <= DN_min or fine0[ib, j, i] >= DN_
max:
                            fine01 = np.sum(weight*(fine1[ib, aj:bj+1, ai:bi+1
]).ravel()[indcand])
                            fine02 = np.sum(weight*(fine2[ib, aj:bj+1, ai:bi+1
]).ravel()[indcand])
                            fine0[ib, j, i] = T_weight1 * fine01 + T_weight2 *
 fine02

                else:  # for the case of no enough similar pixel selected

                    for ib in range(0, nb):
                        # compute the temporal weight
                        difc_pair1 = np.mean((coarse0[ib, aj:bj+1, ai:bi+1]).r
avel()[ind_wind_valid])-
np.mean((coarse1[ib, aj:bj+1, ai:bi+1]).ravel()[ind_wind_valid])+0.01**5
                        difc_pair1_a = np.abs(difc_pair1)
                        difc_pair2 = np.mean((coarse0[ib, aj:bj+1, ai:bi+1]).r
avel()[ind_wind_valid])-
np.mean((coarse2[ib, aj:bj+1, ai:bi+1]).ravel()[ind_wind_valid])+0.01**5
                        difc_pair2_a = np.abs(difc_pair2)
                        T_weight1 = (1.0/difc_pair1_a) / (1.0/difc_pair1_a+1.0
/difc_pair2_a)
                        T_weight2 = (1.0/difc_pair2_a) / (1.0/difc_pair1_a+1.0
/difc_pair2_a)
                        fine0[ib, j, i] = T_weight1 * (fine1[ib, j, i] + difc_
pair1) + T_weight2 * (fine2[ib, j, i] + difc_pair2)

    print('finish ', str(isub + 1), 'block')
    tempoutname1 = temp_file + '\\temp_blended'
    Out_Name = tempoutname1 + str(isub + 1) + suffix
    fp = path1
    writeimage(fine0, Out_Name, fp)

# # ***************************************************************
# # mosaic all the blended patch

datalist = []
minx_list = []
maxX_list = []
minY_list = []
maxY_list = []

for isub in range(0, n_ns * n_nl):
    out_name = temp_file + '\\temp_blended' + str(isub+1) + suffix
    datalist.append(out_name)

    col1 = ind_patch[isub, 0]
    col2 = ind_patch[isub, 1]
    row1 = ind_patch[isub, 2]
    row2 = ind_patch[isub, 3]

    minx_list.append(col1)
    maxX_list.append(col2)
    minY_list.append(row1)
    maxY_list.append(row2)

minX = min(minx_list)
maxX = max(maxX_list)
minY = min(minY_list)
maxY = max(maxY_list)

xOffset_list = []
yOffset_list = []
i = 0
for data in datalist:
    xOffset = int(minx_list[i] - minX)
    yOffset = int(minY_list[i] - minY)
    xOffset_list.append(xOffset)
    yOffset_list.append(yOffset)
    i += 1

in_ds = gdal.Open(path1)
path = os.path.splitext(path5)[0] + "_ESTARFM" + suffix
if suffix == '.tif':
    driver = gdal.GetDriverByName("GTiff")
elif suffix == "" or suffix == ".dat":
    driver = gdal.GetDriverByName("ENVI")
dataset = driver.Create(path, orig_ns, orig_nl, nb, gdal.GDT_Float32)

i = 0
for data in datalist:
    nl, ns, datavalue = read_raster(data)
    for j in range(0, nb):
        dd = datavalue[j, :, :]
        dataset.GetRasterBand(j + 1).WriteArray(dd, xOffset_list[i], yOffset_l
ist[i])
    i += 1

geoTransform = in_ds.GetGeoTransform()
dataset.SetGeoTransform(geoTransform)
proj = in_ds.GetProjection()
dataset.SetProjection(proj)
```

Declaration of consent

on the basis of Article 30 of the RSL Phil.-nat. 18

Name/First Name: Müller Thierry

Registration Number: 16-721-250

Study program: Msc Climate Science

Bachelor ○ Master ● Dissertation ○

Title of the thesis: Assessment of the surface urban heat island in Bern by fusing the Landsat and MODIS LST products

Supervisor: Prof. Dr. Stefan Brönnimann
Prof. Dr. Stefan Wunderle
Dr. Moritz Gubler

I declare herewith that this thesis is my own work and that I have not used any sources other than those stated. I have indicated the adoption of quotations as well as thoughts taken from other authors as such in the thesis. I am aware that the Senate pursuant to Article 36 paragraph 1 litera r of the University Act of 5 September, 1996 is authorized to revoke the title awarded on the basis of this thesis.

For the purposes of evaluation and verification of compliance with the declaration of originality and the regulations governing plagiarism, I hereby grant the University of Bern the right to process my personal data and to perform the acts of use this requires, in particular, to reproduce the written thesis and to store it permanently in a database, and to use said database, or to make said database available, to enable comparison with future theses submitted by others.

Bern, 26.4.2023

Place/Date

Signature