

On remote sensed NDVI and soil moisture correlations

UTRECHT UNIVERSITY

BACHELOR THESIS

Erik Duijm

Student number 4016548

supervised by
Dr. ir. Sander Houweling

July 7, 2016

Contents

1	Abstract	2
2	Introduction	2
3	Data	3
3.1	GIMMS NDVI	3
3.2	ESA CCI Soil moisture	5
3.3	GLC2000	8
4	Method	11
4.1	Preparing the data	11
4.2	Statistical analysis	12
4.3	Transforming GLC 2000	13
4.4	Data filtering	15
5	Results	16
6	Discussion and future work	23
7	Conclusion	24
8	Appendix	26

1 Abstract

This report investigates the drivers of the observed interannual variability in atmospheric CO₂. It is known that much of that variability is caused by the response of the terrestrial biosphere to climatic variations, but the dominant mechanisms are still poorly understood. In a paper by Poulter et al.[1] a significant carbon sink anomaly is described. This anomaly of an approximately 57% increase in carbon uptake, was predominately caused by semi-arid regions. These anomalies point to the important role of the climate sensitivity of semi-arid ecosystems, which had received little attention until then. In order to explain the variability caused by the semi-arid regions we look at the limiting factor for these regions. The availability of water. To see whether or not limited water availability causes a high variability we compare soil moisture deviations with NDVI deviations. NDVI is a measure of the photosynthetic capacity of vegetation. Using ESA CCI Soilmoisture and GIMMS AHVRR NDVI data sets we found a positive Pearson correlation for deviations from the 10 year average of 2002-2012. This correlation was found to have a higher coefficient when a time lag was introduced. The highest coefficients were found when soil moisture data was related with NDVI data measured 16 days later. The areas that showed the highest correlation coefficients were of the semi-arid nature. These areas are thus the most sensitive to soil moisture anomalies and cause a high variability in photosynthetic capacity.

2 Introduction

Our climate is changing and CO₂ emissions play an important role. In order to make long-term climate predictions we need to be able to predict the amount of atmospheric CO₂. For these predictions we need to understand the global carbon cycle. In the global carbon cycle the atmospheric CO₂ and the CO₂ in the oceans and biomass used to be in equilibrium. Usage of fossil fuels has significantly increased the atmospheric CO₂ and caused an imbalance in the global carbon cycle. This atmospheric CO₂ will partially get absorbed by the oceans and the biosphere. There is a lot of uncertainty in long term climate models caused by uncertainty of the climate sensitivity of the absorption of the atmospheric CO₂. Thus the growth of atmospheric CO₂ and the speed of this growth are important for our understanding of the global carbon cycle. We know that the CO₂ growth speed shows important variations and that the El Nino-Southern Oscillation (ENSO) is a critical driver of the variability [2] which can be used to investigate the climate sensitivity of biospheric CO₂ uptake. One of the more conventional theories on this CO₂ growth rate - ENSO relation is that it is driven primarily by tropical rainforests [2]. However, Poulter et al.[1] suggests that semi-arid areas are in fact more important than expected. This implies that rainforests play a smaller role than assumed. Their paper describes the finding of an unusually large carbon land sink in 2011. This anomaly appeared to be caused by a very large La Nina event. Poulter et al. report a increase of approximately 57% in carbon land sink[1], 79 to 87% of this anomaly could explained by the regions Australia, temperate South America and Southern Africa.

Since in these semi-arid regions water is limited we look into this as a possible cause of the observed carbon sink variability. In a attempt to further improve our understanding of the global carbon cycle we study satellite data to investigate the effect of deviations in the availability of soil moisture. We are going to look at whether or not there is a correlation between deviations in soil moisture and the Normalized Difference Vegetation Index(NDVI). NDVI is based on the measurement of the frequency of light, reflected by plants that are photosynthesizing and thus used as an indication of photosynthetic capacity, i.e. the capacity to store carbon and influence the CO₂ growth rate. Since CO₂ and photosynthetic capacity are related we want to use any relation we find to improve our understanding of the variability in CO₂. In order to properly study the effect of soil moisture deviations on NDVI, we introduce a time lag. This lag is used to account for the fact that the vegetation response to an anomaly in soil moisture takes a certain amount of time. The paper by Chen et al.[3] suggests such that a lag does indeed exist. They found that a large precipitation anomaly over Australia caused a soil moisture deviation that influenced

the biomass for a couple of months.

For this study we use the European Space Agencies (ESA) Climate Change Initiative (CCI) soil moisture dataset, a product that contains global soil moisture measurements. Due to the physics of the microwave radiation that is used we can only observe the top few centimeters. As a result, the relationship between soil moisture and NDVI is influenced by the sensitivity of vegetation to water availability in the surface layer, which depends on the rooting depth of the vegetation. This could mean that we find no relation at all, due to plants not being dependent on the availability of top layer soil moisture. However, the question we are looking into is whether or not we can use this data to learn something about the semi-arid regions. These regions are often dominated by vegetation types, such as grasses and shrubberies, with limited root depth.

Our hypothesis is that deviations in soil moisture and deviations in NDVI are positively correlated. We also expect that introducing a lag increases any relation we find. In this paper we will first discuss the specifications of the data we used. Then we will explain the methods we used to derive our results. The result are followed by a general discussion leading to conclusions and recommendations for follow on research.

3 Data

In the following section we discuss the data used for our research. We will explain for each dataset the specifications of the data, the quality of the data and the reason we chose a specific product. We will introduce each data set by explaining the theory behind the data, and how they fit into our research.

3.1 GIMMS NDVI

We used the Global Inventory Modeling and Mapping Studies (GIMMS) satellite product[4]. This product contains the Normalized Difference Vegetation Index (NDVI), derived from radiance measurements from the Advanced Very High Resolution Radiometer (AVHRR) instrument aboard several NASA satellites. The first of these satellites was the TIROS-N, launched in 1978. Ever since then the AVHRR was improved and new satellites were launched. We have data available since 1981, when NOAA-7 was launched with an improved AVHRR compared to the one aboard TIROS-N. The most recent version of the AVHRR is aboard NOAA-15 launched in 1998[5]. Thanks to AVHRR instruments aboard several satellites[6], launched in a sequence of satellite missions, we have a relatively long dataset available, namely from 1981 to 2013. NDVI makes use of the spectral characteristics of photosynthesizing vegetation. The green leaves of plants strongly absorb visible light (wavelength in the range 0.4 to 0.7 μm , excluding the green visible light), while the leaves strongly reflect in the near infrared light (0.7 to 1.1 μm). Since the AVHRR can measure these different wavelengths of light we can use this to define NDVI:

$$NDVI = \frac{NIR - VIS}{NIR + VIS} \tag{1}$$

where NIR is the measured light at near infra red wavelengths, and VIS stands for the light measured at visible wavelengths. We can see that this gives us a measure for vegetation greenness that ranges from -1 to 1. Dense photosynthesizing vegetation thus reflects a lot of light in the near infra red, and absorbs a large amount of visual light. Thus a high NDVI represents a high photosynthetic capacity. In figure 1 we can see an example of what a NDVI data set looks like. The figure represents a global map of NDVI measured during August 2009. The dataset defines oceans to have an NDVI of -1.

The dataset we used has a resolution 1/12 degree by 1/12 degree. The data is averaged over 16 days by NASA. The quality of the data is indicated by four different flags. These flags tell us if the data is acceptable, whether or not snow influenced the observations and tells us about the type of interpolation used, if any. The uncertainty of the GIMMS NDVI data has been evaluated by Tucker et al. [7] who

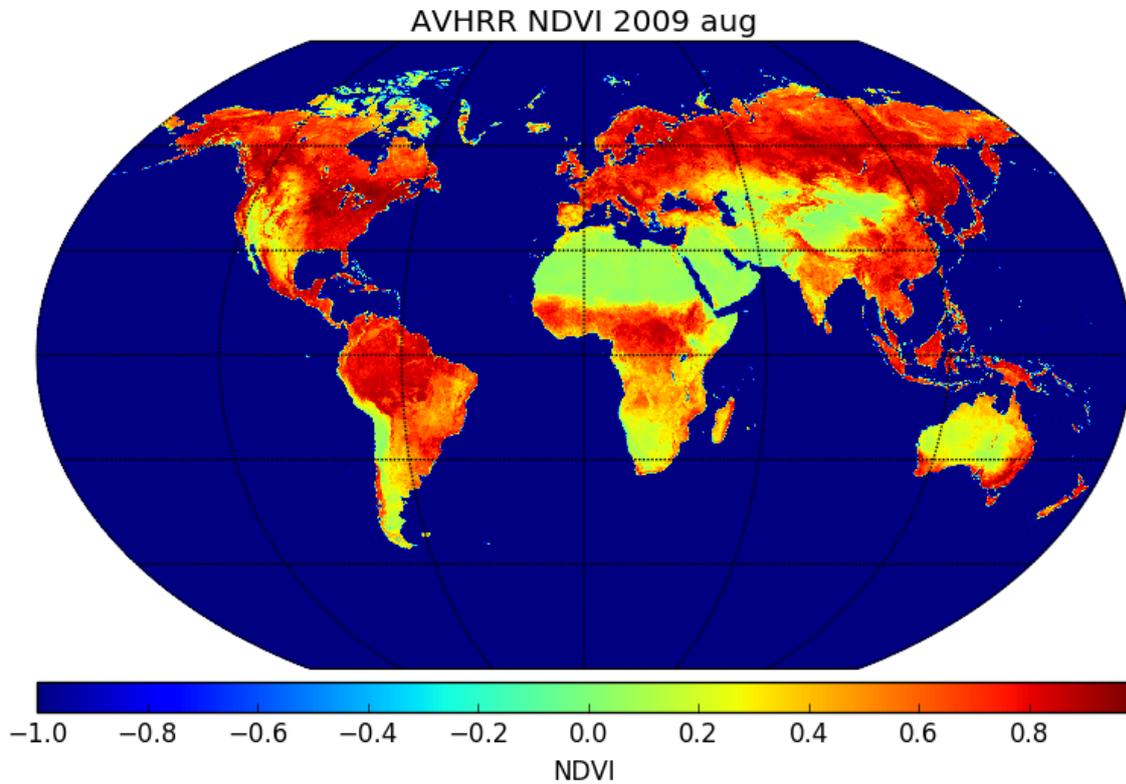


Figure 1: Example of GIMMS NDVI data. Image created using the data from August 2009.

report a uncertainty of ± 0.005 NDVI units. This reported error is a conservative estimate based on estimates of the error in spatial, seasonal and temporal coherence. There are several reasons we chose the GIMMS AVHRR NDVI product over other NDVI products. The data was publicly available and easy to download. The size of the data was relatively small and therefore easy to handle. It is also one of the more widely used products and thus easily comparable with other research, since many years of data are available.

3.2 ESA CCI Soil moisture

The soil moisture data set we used is part of ESA's Climate Change Initiative (CCI)[8][9][10]. This data is produced by ESA CCI soil moisture group and is created out of measurements of several satellites. Figure 2 presents an overview of the satellites used and the time period during which they were active.

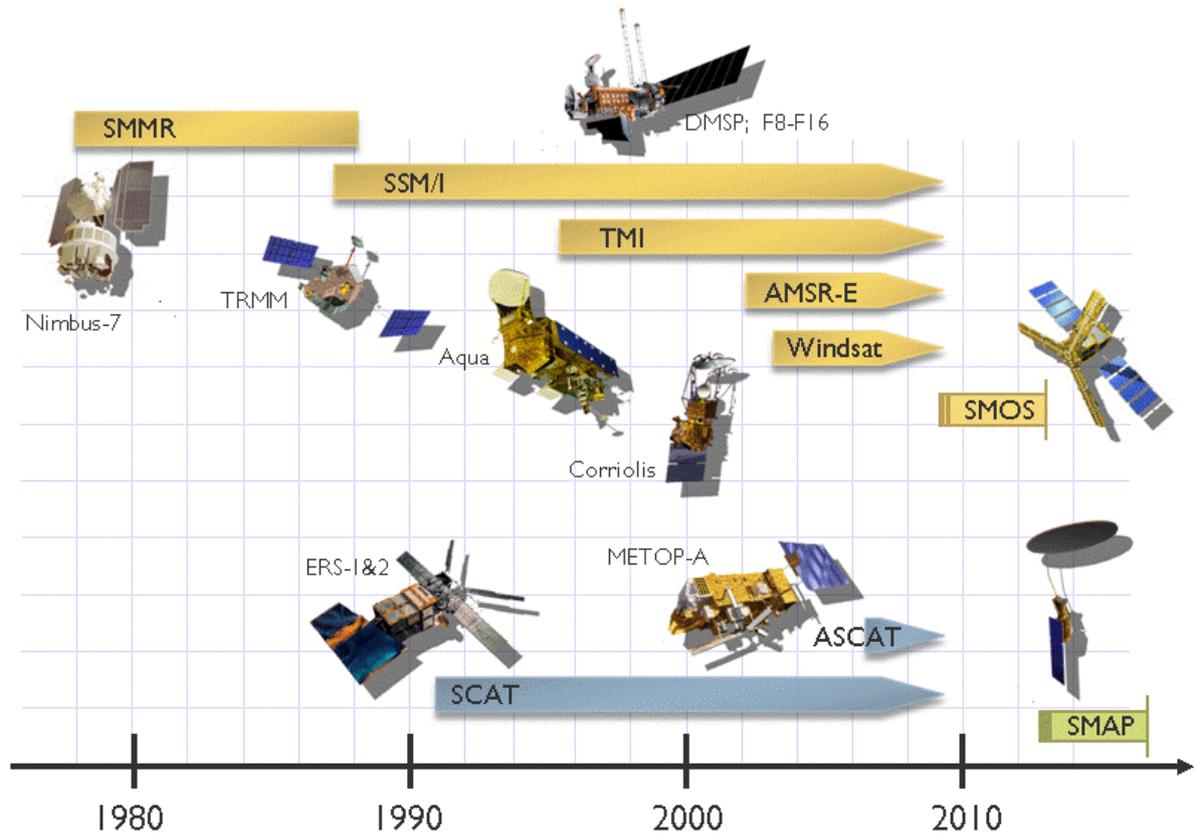


Figure 2: An overview of satellites used by ESA to measure soil moisture. Image taken from <http://www.esa-soilmoisture-cci.org/node/93>

These satellites use several instruments to measure the soil moisture in the top few centimeters [11]. In figure 3 we can see an example of the soil moisture data for August 2009. To create this image we have averaged the data over 16 days. We did this to ensure global coverage and to make it comparable with the NDVI data. For comparison we also show the data of just a single day, August 16 2009. In this figure we can clearly see gaps in the coverage and confirm that we need some degree of averaging to ensure global coverage. The gaps in the coverage are caused by limits on what the satellite can cover in a day and when the retrieval software encountered conditions for which it can not generate valid measurements[12].

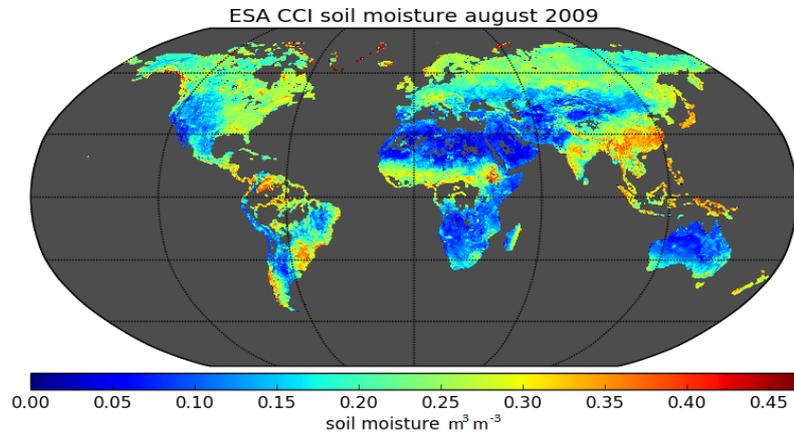


Figure 3: Example of soil moisture data. Image created using the data from August 2009.

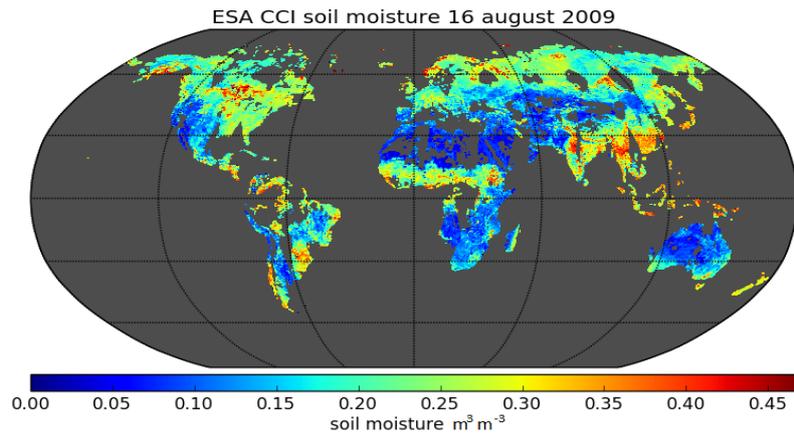


Figure 4: Example of a single day of soil moisture data. Image created using the 16 August 2009 data.

The ESA CCISM data comes at a 0.25 by 0.25 degree resolution, in time intervals of a day. The data is available from 1978 to 2014. The ESA CCISM product contains three different data subsets. An active microwave, passive microwave and combined dataset. The active microwave dataset is derived from measurements from light actively emitted by a radar antenna on board the satellite, which is then reflected at the surface. The passive microwave product contains data derived from light emitted by the earth's surface. We use the combined product as it couples the active and passive datasets[12]. This dataset contains a scaled soil moisture value. The values have unit $m^3 m^{-3}$ with a scaling factor of 0.0001. The product provides an uncertainty of the measurements. We take the upper bound in these

uncertainties to be $0.12 m^3 m^{-3}$ for the data we are using. Again the quality of the data is described with flags provided by the product. There are four different flags specifying: good data, dense vegetation, snow and freezing temperatures, and invalid data. The flags for dense vegetation and snow were added since these circumstances prevent proper observations of the ground. We have only used soil moisture data from 2002 and later. The reason is that in 2002 ESA launched a new satellite that greatly improved the quality and coverage of the data[12].

3.3 GLC2000

The Global Land Cover (GLC) 2000 project is part of the European Commission's program called Global Environment Information System[13]. The GLC 2000 is a project that created a map of the global ecosystems. This map was created by 30 teams, who looked at 19 regional windows. In figure 5 we see these different areas.

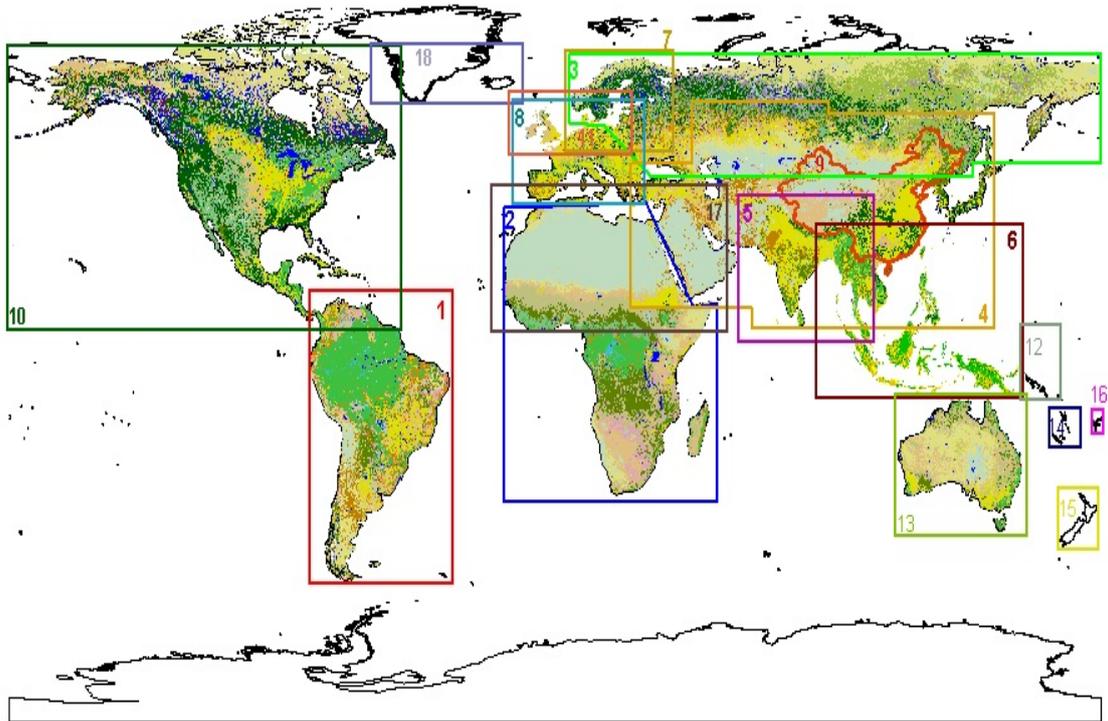


Figure 5: Figure showing the regional windows used to create the GLC map. Image taken from <http://forobs.jrc.ec.europa.eu/products/glc2000/products.php>

For each of these regions the teams working on them were allowed to use their own methods to map their areas. This was done to ensure quality, since the regional experts had a good understanding of their region. These different maps were then combined to the global product that can be seen in the image 6 below. This was done by assigning all the different ecosystems into classes that can be seen in table 1 which also serves as legend for figure 6.

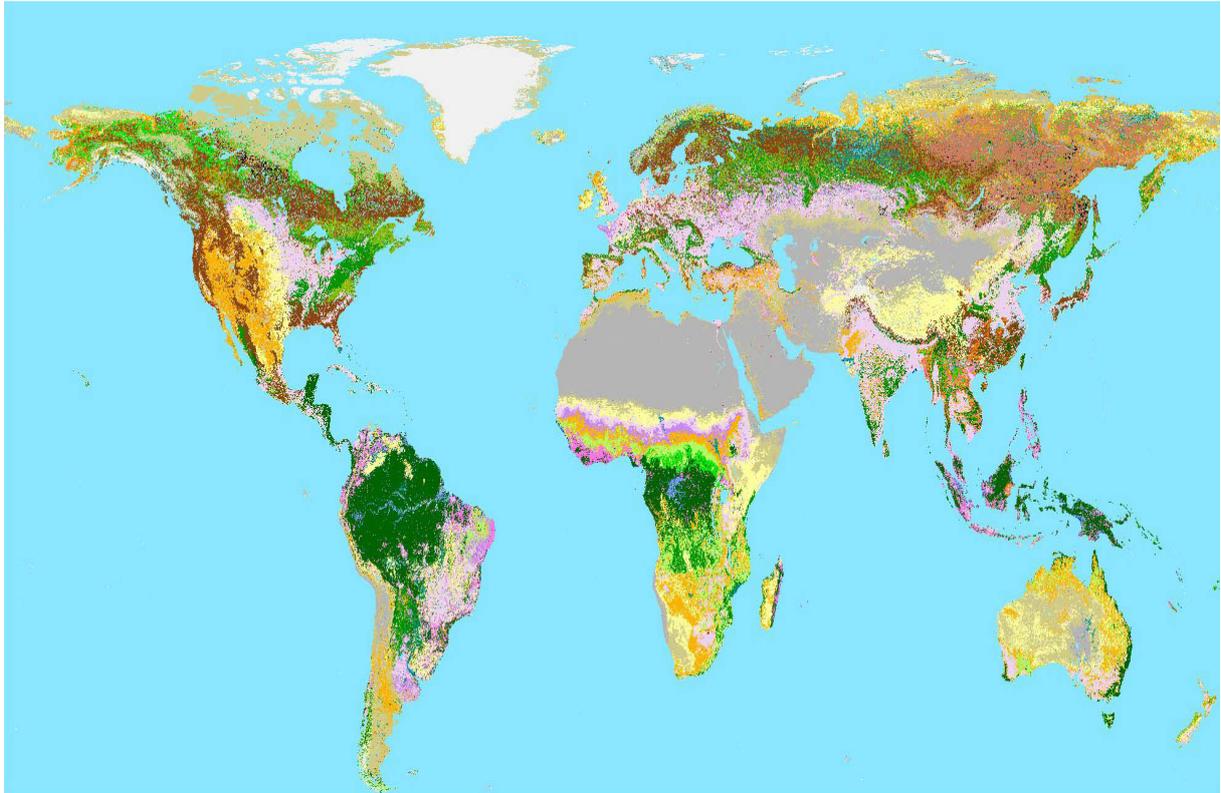


Figure 6: Global landcover map. Each color represents an ecosystem. See table 1 for the legend.

GLC Global Class (according to LCCS terminology)
<ul style="list-style-type: none"> Tree Cover, broadleaved, evergreen <i>LCCS >15% tree cover, tree height >3m</i> (Examples of sub-classes at regional level* : <i>closed > 40% tree cover; open 15-40% tree cover</i>)
<ul style="list-style-type: none"> Tree Cover, broadleaved, deciduous, closed
<ul style="list-style-type: none"> Tree Cover, broadleaved, deciduous, open (<i>open 15-40% tree cover</i>)
<ul style="list-style-type: none"> Tree Cover, needle-leaved, evergreen
<ul style="list-style-type: none"> Tree Cover, needle-leaved, deciduous
<ul style="list-style-type: none"> Tree Cover, mixed leaf type
<ul style="list-style-type: none"> Tree Cover, regularly flooded, fresh water (& brackish)
<ul style="list-style-type: none"> Tree Cover, regularly flooded, saline water, (daily variation of water level)
<ul style="list-style-type: none"> Mosaic: Tree cover / Other natural vegetation
<ul style="list-style-type: none"> Tree Cover, burnt
<ul style="list-style-type: none"> Shrub Cover, closed-open, evergreen (Examples of sub-classes at reg. level* : (i) sparse tree layer)
<ul style="list-style-type: none"> Shrub Cover, closed-open, deciduous (Examples of sub-classes at reg. level* : (i) sparse tree layer)
<ul style="list-style-type: none"> Herbaceous Cover, closed-open (Examples of sub-classes at regional level* : (i) natural, (ii) pasture, (iii) sparse trees or shrubs)
<ul style="list-style-type: none"> Sparse Herbaceous or sparse Shrub Cover
<ul style="list-style-type: none"> Regularly flooded Shrub and/or Herbaceous Cover
<ul style="list-style-type: none"> Cultivated and managed areas (Examples of sub-classes at reg. level* : (i) terrestrial; (ii) aquatic (=flooded during cultivation), and under terrestrial: (iii) tree crop & shrubs (perennial), (iv) herbaceous crops (annual), non-irrigated, (v) herbaceous crops (annual), irrigated)
<ul style="list-style-type: none"> Mosaic: Cropland / Tree Cover / Other natural vegetation
<ul style="list-style-type: none"> Mosaic: Cropland / Shrub or Grass Cover
<ul style="list-style-type: none"> Bare Areas
<ul style="list-style-type: none"> Water Bodies (natural & artificial)
<ul style="list-style-type: none"> Snow and Ice (natural & artificial)
<ul style="list-style-type: none"> Artificial surfaces and associated areas

Table 1: Global land coverage legend.

The global map has 1 km spatial resolution, and describes the land cover of the year 2000.

4 Method

In this section we explain the steps that were taken to analyse the datasets described in section 3. We will explain how we analyze the lag and how we calculate correlations.

4.1 Preparing the data

The first step to compare the datasets is to define a standard resolution and timescale. For resolution we chose the lowest resolution available. The reason behind this is that we can not increase that resolution without additional information. However, a higher resolution can be lowered by averaging. The lowest resolution is the 0.25 (28 km) degree resolution of the ESA CCISM dataset. Thus we change the resolution of our NDVI data to 28 km by averaging. Since the NDVI dataset has a 8 km resolution with 4320 by 2160 data points each point of the new data set, with 1440 by 720 data points consists of the average of the nine data points that fit in the lower resolution. 1440 by 720 data points is the ESA CCISM format. After the data sets were brought to a common spatial resolution we need to harmonize the timescale. Since the NDVI data is averaged over 16 days by NASA, this is the highest common temporal resolution. This also implies that all derived timescales can only be multiples of 16 days. To ensure that we have the best possible data to calculate correlations, we chose the 16 day timescale, since this is the highest possible temporal resolution. Since the soil moisture data is daily it was averaged to the 16 day timescale. To ensure the quality of the data we only use the data that has been flagged as good data in both datasets. Since our interest lies with the sensitivity to variability we need to look at the response NDVI has to a change in soil moisture. In order to know what an anomaly is we need a normal value to compare with. To create a baseline, we averaged the data for both NDVI and soil moisture over the period 2002-2012. We created monthly averages so that we could remove any seasonal cycle in our data. We did this by subtracting the average seasonality from the data. These anomalies then have no seasonal effects, since they are deviations from what is normal in that season.

With these averages we can calculate the deviations in our data, and start relating these. To ensure that we are looking at significant deviations we set a threshold. We require that there must be significant deviations (at least 0.01 NDVI units or $0.01\text{m}^3\text{m}^{-3}$). This is done for the interpretation of correlations in areas with no significant anomaly. The value 0.01 was chosen as there was no obvious cutoff point in deviations as can be seen in figure 7, and a boundary larger then the data uncertainty was required as will be discussed in the statistical analysis section. We can also see from the image that by doing this we are not discarding a significant fraction of our data. Using this method for the data from 2009 14.2% of NDVI anomalies and 4.3% of soil moisture anomalies is discarded.

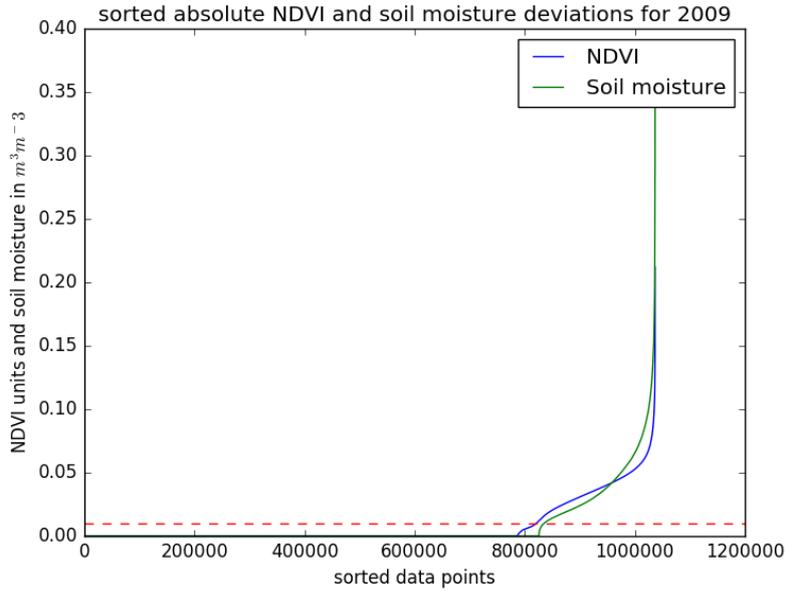


Figure 7: Sorted absolute NDVI and soil moisture deviations. The dotted line represents 0.01. The values shown are averaged, per location, over the period 2002-2012.

4.2 Statistical analysis

Now that we have our deviations we can start looking at relations. We compared the soil moisture and NDVI anomalies using the Pearson correlation. The Pearson correlation is calculated using the formula:

$$r = \frac{\sum_i^n (x_i - \bar{x})(y_i - \bar{y})}{(n-1)\sigma(x)\sigma(y)} \quad (2)$$

where x and y are the variables being correlated and the bar indicates the average of that variable. $\sigma(x)$ indicates the standard deviation of variable x .

Using this formula we get a Pearson correlation coefficient, r , that ranges from -1 to 1. This number is a measure of the linear relation between variables. A positive value indicates that x and y are linearly related and a negative value stand for an linear relation with a negative slope. A zero indicates that there is no linear relation. This does however not exclude there being any non linear relation.

If we want to take a look at the presence of time lag we need to correlate time series. We do this by relating data over the time span of a year. This gives us 24 data points for each coordinate. An example is given in figure 8. To introduce lag we create a shift in the input data and thus, for example, are relating the soil moisture data with the NDVI data from 16 days later.

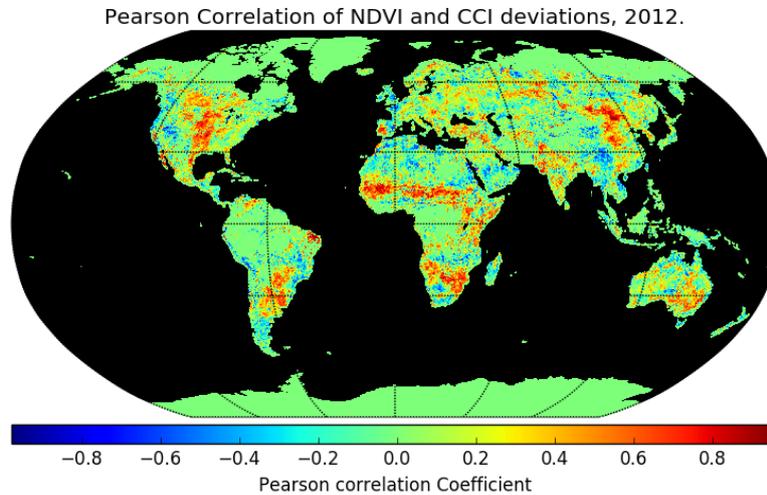


Figure 8: Example of the Pearson correlation map. Map show the results for 2012 with no lag.

4.3 Transforming GLC 2000

Next we can compare the created map of correlations with the land coverage map to investigate the relation between correlation and ecosystem type. To do this we first need to downscale the global land cover map since it has a 1 km spatial resolution, where as our correlation map will have a 28 km resolution. In order to say something about the Pearson correlation result in relation to specific ecosystems we need to work in the same resolution. However the GLC data is discrete, and thus it can't simply be averaged. In order to map the GLC map to the 0.25 degree resolution we apply the following rule: If at least 75% of the land coverage is of a single ecosystem class we assign that class to the entire lower resolution grid box.

Due to the specific classifications of the GLC map this requires us to introduce a more general legend. The new legend can be seen in table 2. In figure 10 we can see the result of introducing our own classifications. For comparison, it is shown what the GLC map would have looked like if we did not introduce our own classification. It should be noted that the color red that dominates figure 9 is added to indicate an area where not more then 75% was of a single ecosystem type.

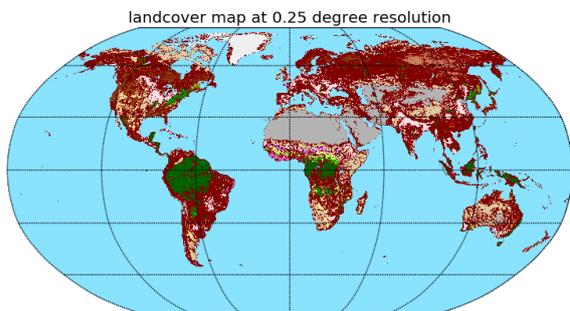


Figure 9: GLC at 28 km resolution

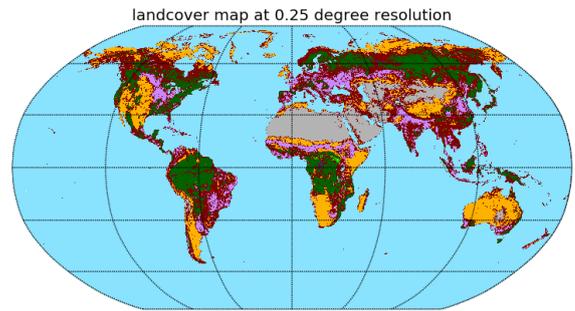


Figure 10: GLC at 28 km using the new classification in the legend from table 2

GLC Global Class (according to LCCS terminology)	GLC Global Reclassification
<ul style="list-style-type: none"> Tree Cover, broadleaved, evergreen <i>LCCS >15% tree cover, tree height >3m</i> (Examples of sub-classes at regional level* : <i>closed > 40% tree cover; open 15-40% tree cover</i>) Tree Cover, broadleaved, deciduous, closed Tree Cover, broadleaved, deciduous, open (<i>open 15-40% tree cover</i>) Tree Cover, needle-leaved, evergreen Tree Cover, needle-leaved, deciduous Tree Cover, mixed leaf type Tree Cover, regularly flooded, fresh water (& brackish) Tree Cover, regularly flooded, saline water, (daily variation of water level) 	<p>Ecosystem dominated by tree cover</p>
<ul style="list-style-type: none"> Mosaic: Tree cover / Other natural vegetation Tree Cover, burnt 	<p>Mosaic: Tree cover / Other natural vegetation</p> <p>Tree Cover, burnt</p>
<ul style="list-style-type: none"> Shrub Cover, closed-open, evergreen (Examples of sub-classes at reg. level* : (i) sparse tree layer) Shrub Cover, closed-open, deciduous (Examples of sub-classes at reg. level* : (i) sparse tree layer) Herbaceous Cover, closed-open (Examples of sub-classes at regional level* : (i) natural, (ii) pasture, (iii) sparse trees or shrubs) Sparse Herbaceous or sparse Shrub Cover 	<p>Shrub covered areas, low tree density</p>
<ul style="list-style-type: none"> Regularly flooded Shrub and/or Herbaceous Cover 	<p>Regularly flooded area</p>
<ul style="list-style-type: none"> Cultivated and managed areas (Examples of sub-classes at reg. level* : (i) terrestrial; (ii) aquatic (=flooded during cultivation), and under terrestrial: (iii) tree crop & shrubs (perennial), (iv) herbaceous crops (annual), non-irrigated, (v) herbaceous crops (annual), irrigated) Mosaic: Cropland / Tree Cover / Other natural vegetation Mosaic: Cropland / Shrub or Grass Cover 	<p>Cultivated and managed areas</p>
<ul style="list-style-type: none"> Bare Areas 	<p>Bare Areas</p>
<ul style="list-style-type: none"> Water Bodies (natural & artificial) Snow and Ice (natural & artificial) 	<p>Water bodies and snow and ice regions</p>
<ul style="list-style-type: none"> Artificial surfaces and associated areas 	<p>Artificial surfaces and associated areas</p> <p>Area with less than 75% of a single ecosystem class</p>

Table 2: Global land coverage legend and the new classification.

For our classification we made a crude distinction between areas dominated by trees, and more shallow rooted vegetation. This was done since the ESA CCISM data is limited to the top few centimeters of

soil, and thus expected root depth to be an important limiting factor for the resulting correlations.

4.4 Data filtering

To be able to conclude anything from our results we need to look at the uncertainty of the data, and how it propagates in our calculations. For this we take the upper bound of the uncertainty ranges reported for our datasets. For NDVI this is 0.005 NDVI units. We do however not know how much of this is a random error, and how much is a systematic error. For simplicity we assume that the error is fully random. We know from basic statistics that averaging a random error will lower the error by a factor \sqrt{n} , where n is the number of values being averaged. By changing the resolution we averaged over 9 values, reducing the error by a factor 3. This results in an error of ± 0.00167 .

For soil moisture we use an upper bound of $0.12 \text{ m}^3 \text{ m}^{-3}$ for our uncertainty. We again make the assumption that this error is random, and not systematic. Figure 11 shows the mean uncertainty of a single measurement, averaged over a year.

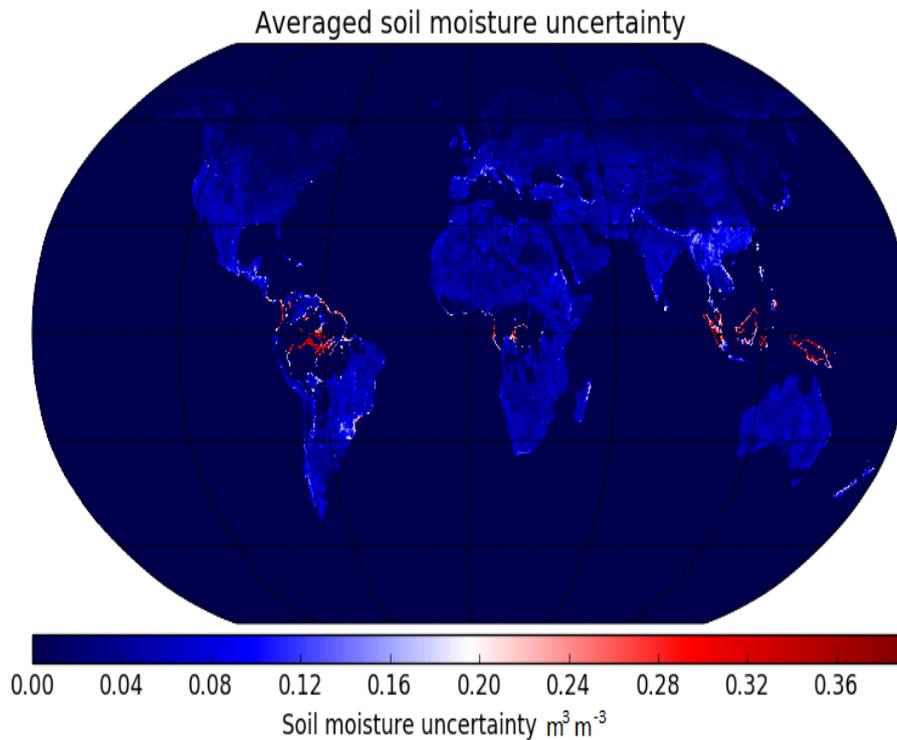


Figure 11: The reported uncertainty of soil moisture measurement averaged over a year.

The uncertainty is lowered by a factor 4 since we are averaging over 16 days to reach our 16 day temporal resolution. This results in an error of $0.03 \text{ m}^3 \text{ m}^{-3}$. Looking at figure 7 we see that there is a

significant portion of soil moisture deviations larger than $0.03 \text{ m}^3\text{m}^{-3}$. We are therefore confident that data uncertainty has only a minor influence on the correlation between NDVI and soil moisture presented in the results. From figure 11 it is clear that tropical regions have an uncertainty higher than the upper bound of $0.12 \text{ m}^3\text{m}^{-3}$, therefore we can not make any conclusions about these regions because any correlation we find there might be caused by an error in our soil moisture data.

5 Results

Using the procedure described in the methods section we have calculated a Pearson correlation for each year of data in the period 2002 - 2012. We have calculated the correlation with a lag 16, 32, 0 and -16 days. The figures below show these correlations, averaged over the 10 year period, for each of the different lags.

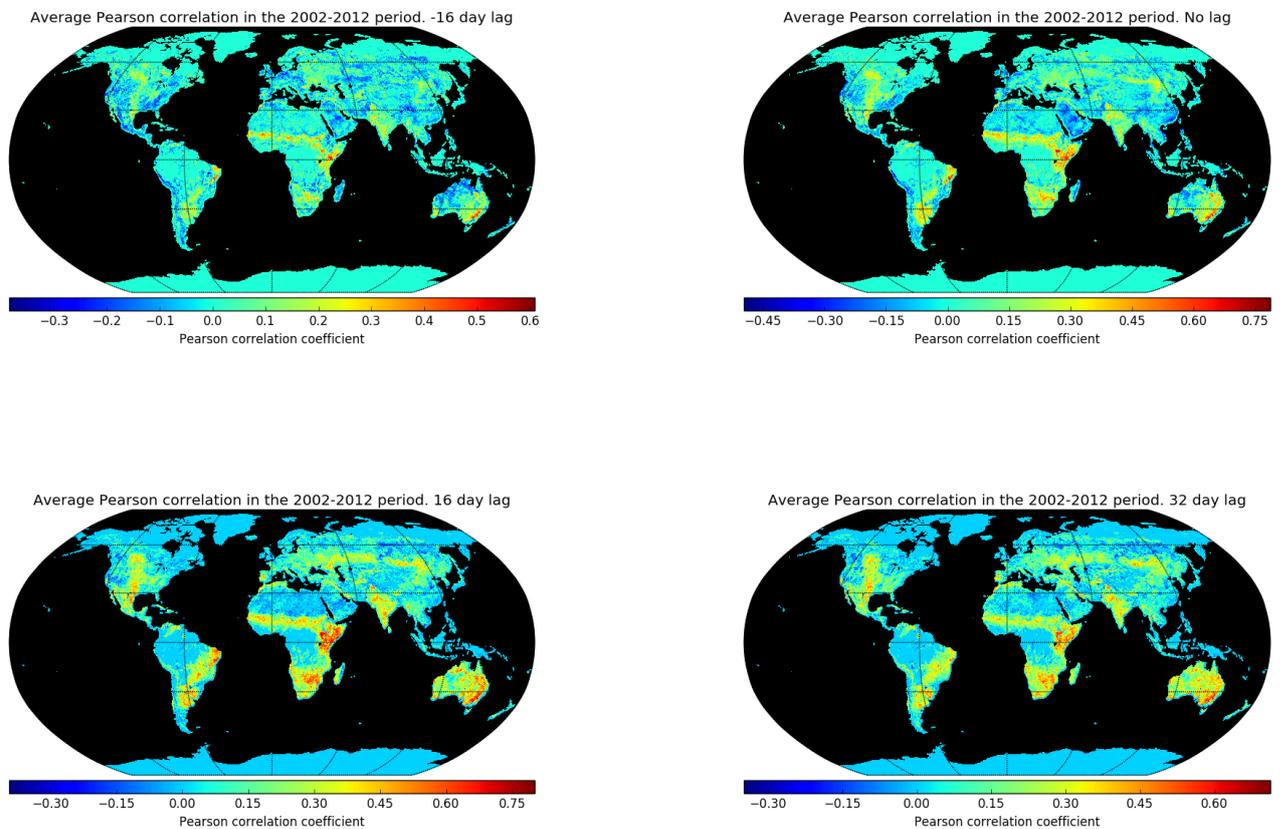


Figure 12: Figures showing averaged Pearson correlation coefficients for -16, 0 ,16 and 32 day lag

In these images we can clearly see how the correlation varies with geographical region and lag time. In order to compare the level of correlation and effect of the different amounts of lag we have plotted the globally averaged correlation coefficients in figure 13. In these averages the oceans were excluded.

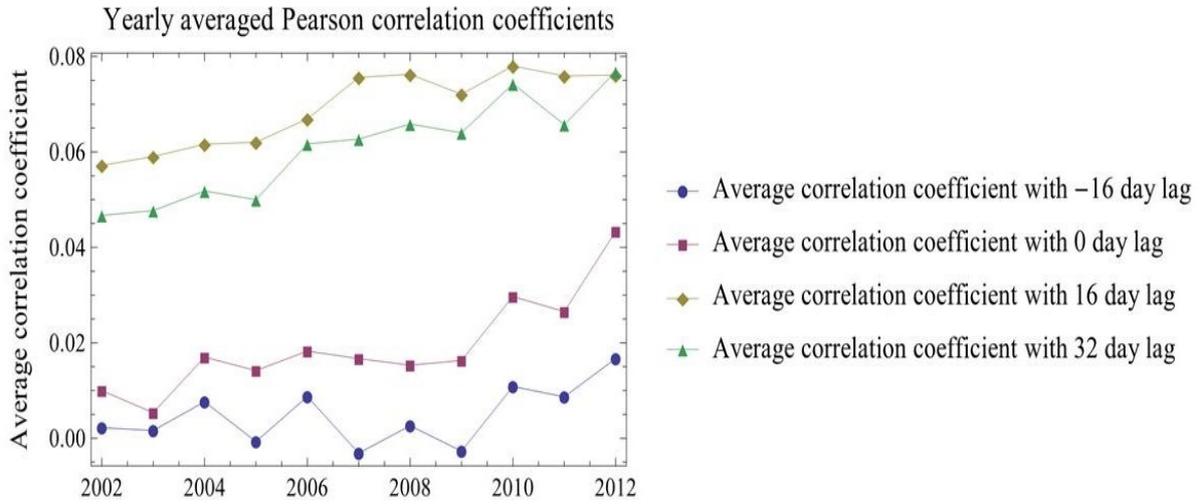


Figure 13: The averaged Pearson correlation coefficients. Plotted for the different lag configurations.

Figure 13 shows that a positive lag generally increases the correlation between NDVI and soil moisture deviations, whereas a negative lag time on average decreases the correlation coefficient.

To see what areas are more effected by particular lag effects we look at the difference in correlation between the different lag times. These can be seen in figures 14, 15 and 16. At first the difference between no lag and 16 day lag is looked at. This can be seen in figure 14. This image shows in red where the correlation is higher for the 16 day lag.

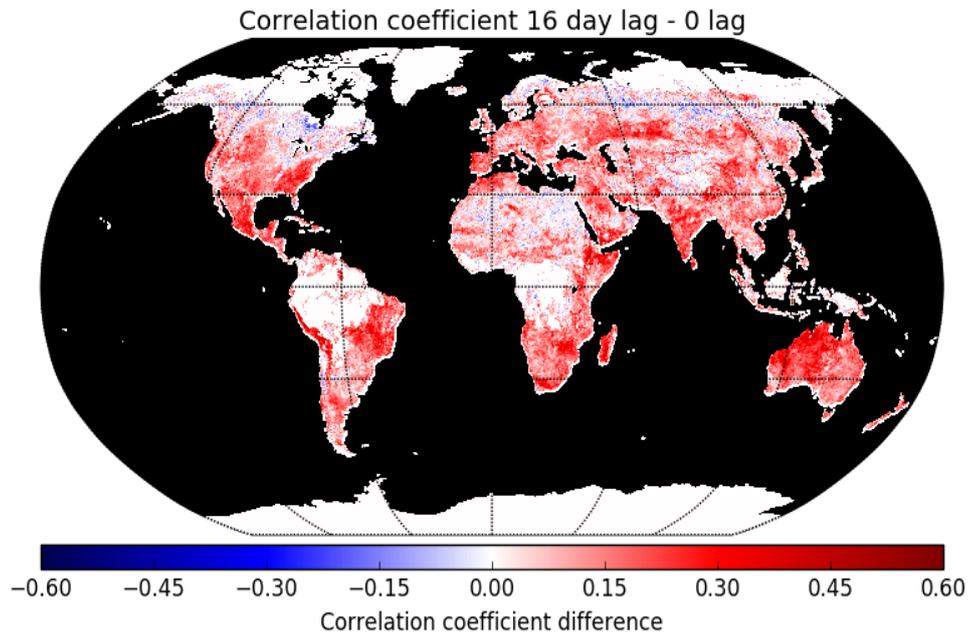


Figure 14: Difference in Pearson correlation coefficient. The values without lag have been subtracted from the corresponding values with 16 day lag.

We see that all significant differences with the results without lag are positive. Thus we observe a higher Pearson correlation coefficient for all areas when compared to the results without lag. To see what areas show a longer lag we plotted the result of the 32 day lag results minus the 16 day lag results in figure 15.

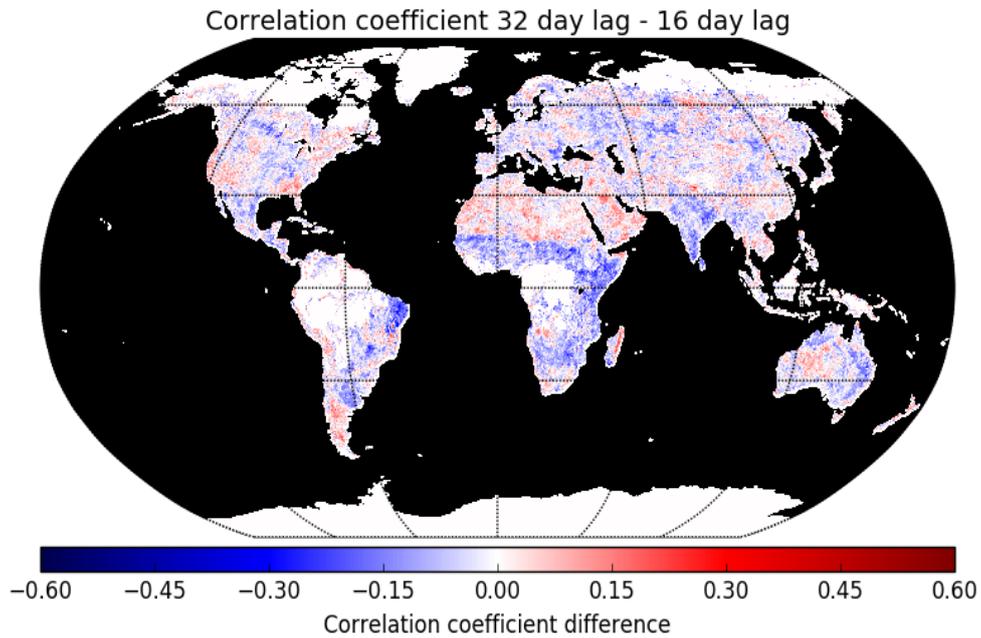


Figure 15: Difference in Pearson correlation coefficient. The values with 16 day lag have been subtracted from the corresponding values with 32 day lag.

From the comparison in figure 15 we can see that in most areas, the 16 day lag shows the strongest correlation. There are a few areas that show a higher correlation with the longer lag. For these areas it appears that an soil moisture anomaly has either a longer, or a more delayed effect on vegetation. As a consistency check we look into the reversed effect, an NDVI anomaly that would appear to cause an soil moisture anomaly, shown in figure 16. In this figure we have plotted the difference of no lag, and a lag of -16 days.

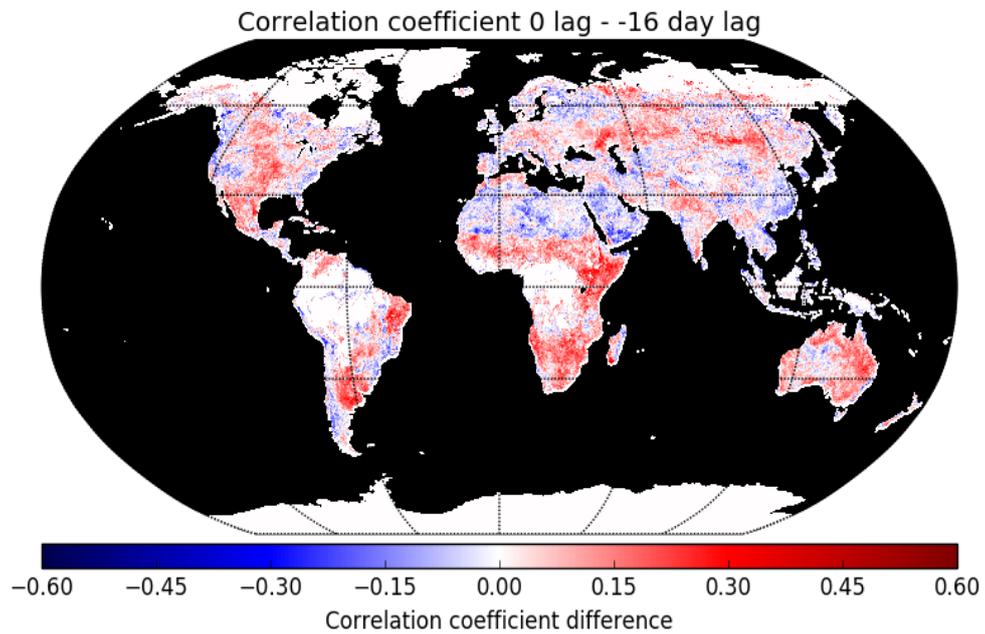


Figure 16: Difference in Pearson correlation coefficient. The values with -16 day lag have been subtracted from the corresponding values without lag.

In order to relate correlation differences to ecosystem types we compare them with the 0.25 degree land coverage map using the reclassification introduced in table 2. In figure 17 all the areas with a correlation coefficient of 0.3 or higher for the 16 day lag results are shown.

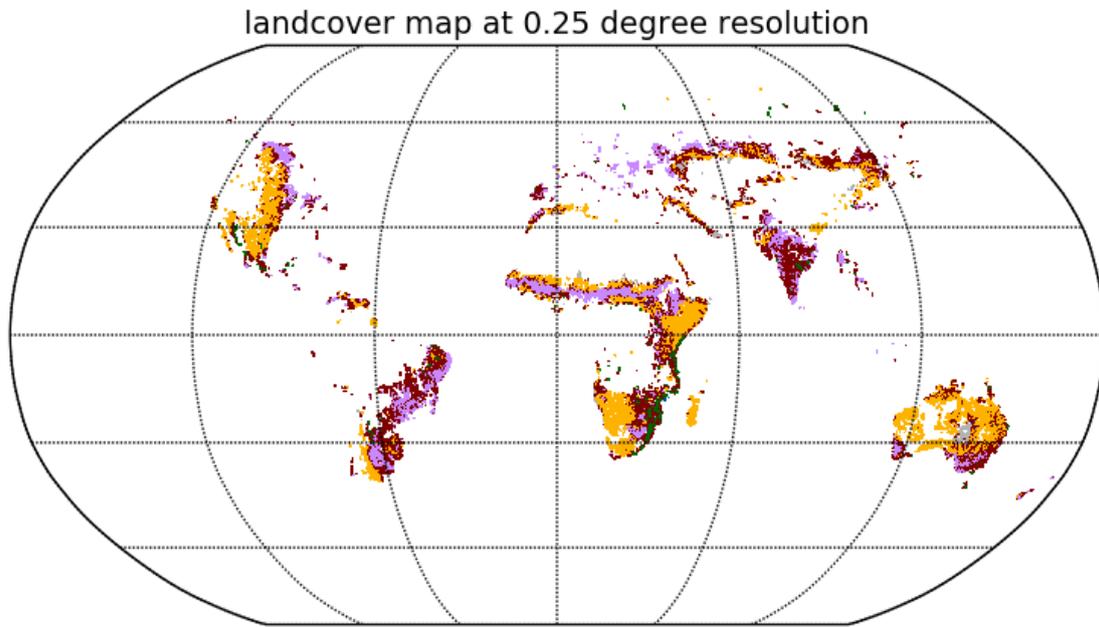
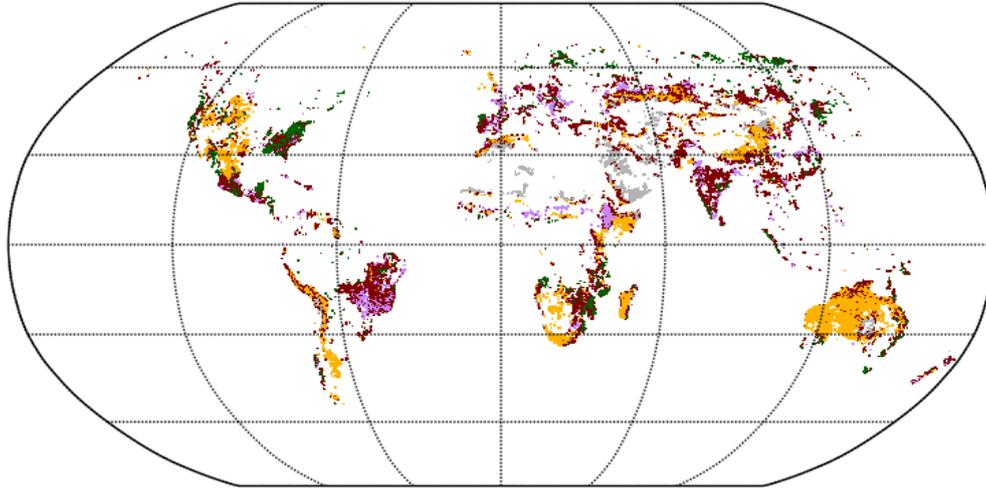


Figure 17: Areas where the correlation coefficient of the 16 day lag data is 0.3 or higher

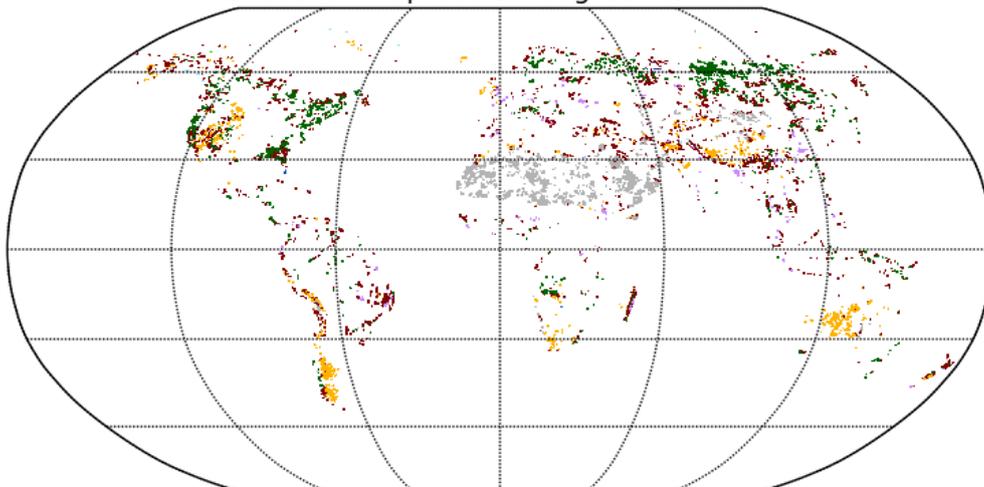
Figure 17 shows that the areas with a significant correlation (coefficient of 0.3 or higher) are almost exclusively the semi-arid and managed and cultivated regions. For both 16 days lag compared to no lag, and 32 day lag compared to 16 day lag, we have made the same comparison to the GLC map. These can be seen in figure 18. This shows the ecosystem classes that are more affected by different types of lag.

landcover map at 0.25 degree resolution



(a) Areas where the correlation coefficient of the 16 day lag data is at least 0.2 higher then data without lag.

landcover map at 0.25 degree resolution



(b) Areas where the correlation coefficient of the 32 day lag data is at least 0.1 higher then the 16 day lag data.

It would appear that there are a few specific areas such as the middle of Australia, and the semi-arid region of North-America show a larger correlation for a 32 day lag, but there is no clear specific ecosystem class shows a higher correlation for the 32 day lag. If we look at 18a, where we took the difference to be at least 0.2, we see that these mostly semi-arid and other low tree density areas such as the managed and cultivated land display a significantly stronger correlation at 16 day lag.

6 Discussion and future work

In our investigation we are very limited in our precision due to the resolution of the data. Since a global approach was taken this resulted in having to take datasets with limited spatial and temporal resolution. By using the GIMMS NDVI dataset we were limited to working with 16 day steps. Working with a daily temporal resolution could result in finding the exact time lag and might provide insight to the exact response time an ecosystem has to an anomaly in soil moisture. Using the ESA CCI soil moisture data limited us to working with a 28 km spatial resolution. Since the NDVI and GLC datasets have a higher spatial resolution this meant losing information. For the GLC dataset especially since this required regriding the data. The 75% rule is an arbitrary percentage. There is no standard for these situations, so in order to make the GLC map comparable this rule was made up. For the creation of the original GLC 2000 map this percentage depended on the ecosystem class and could be as low as 50% [14]. So the 75% rule might be precise enough, but a system where the rule depends on the dominating ecosystem class should be looked into.

The changing of the GLC spatial resolution also required a new legend for the GLC map, where ecosystem classes were combined on the assumption that the roots would be of similar depth. We conclude that the GLC map used only gives an approximate indication of the ecosystem in the area. In order to make a significant comparison to the GLC data, datasets with a higher spatial resolution for NDVI and soil moisture should be used.

In the statistical analysis section the filtering was explained to be 0.01. This was done to exclude insignificant deviations. However the error in CCISM was found to be 0.03. This noise could degrade a portion of the correlations that were found. Since a significant part of the CCISM data has a deviation higher then 0.03 this means that this possible error only effects a small part of our correlations.

The uncertainty used for the CCISM dataset is reduced assuming that there is good data for 16 days. This assumption is incorrect since there are days where some regions have no data. A more precise investigation of the data uncertainty should be conducted. It should be noted that for both the NDVI and soil moisture data the errors were random errors. If they are partially systematic this would increase the reported uncertainty. So in order to make accurate conclusions more information about the uncertainty is required.

The results that are obtained should be used as an indication that there is a correlation between soil moisture deviations and NDVI deviations in semi-arid and managed regions, that this correlation is stronger for these regions than forested ecosystems and that adding lag increases the correlation. We have shown that relating remote sensed soil moisture data with NDVI data is viable. Further research with more precise data should be conducted to find a correlation with high precision. This would allow for a better understanding of the underlying processes. This could soon be a possibility with the more precise Moderate Resolution Imaging Spectroradiometer (MODIS) data. This will provide very high resolution NDVI data. With this higher resolution there are a few possible interesting continuations of our work: Looking at specific regions with high precision and using the relations we find to improve current carbon cycle and CO₂ variability models. Other interesting work that doesn't require higher precision data is looking at a longer period as we did for figure 13 and see whether or not there is a(n) (ENSO) pattern. A more practical application would be using the fact that three is a lag for small scale predictions that could be interesting for farmers for example. The work that we've done has a lot of uncertainties that require more work, but show promise for future work.

7 Conclusion

We used ESA's CCI soil moisture data and NASA's GIMMS AVHRR NDVI to look at the relation between anomalies in photosynthetic capacity and soil moisture. We have seen that there is indeed an observable relation using remote sensing. We have seen that specifically semi-arid regions and the managed areas (e.g. croplands, grasslands) are very sensitive to changes in the availability of soil moisture, and find a correlation coefficient of at least 0.3. These regions also show a stronger correlation (most semi-arid regions show an increase in correlation coefficient of at least 0.2) when a lag is introduced. This time lag is indicative that the underlying process requires that the plants have time to react to a deviation in soil moisture. We have thus confirmed our hypothesis, that NDVI and soil moisture deviations are positively correlated and that lag will increase this correlation. However the positive correlation between NDVI and soil moisture deviations was only found for the semi-arid regions and some of the managed areas. We have shown that relating NDVI and soil moisture remote sensed data for research is possible. Our results confirm that soil moisture measurements from space are potentially useful for studying the global carbon cycle, something which has received relatively limited attention up to now.

References

- [1] Poulter et al, 2014, Contribution of semi-arid ecosystems to interannual variability of the global carbon cycle. *Nature*, 509.
- [2] Wang W. et al, 2013, Variations in atmospheric CO₂ growth rates coupled with tropical temperature. *Proceedings of the National Academy of Sciences of the United States of America* , 110, 13061-13066. <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3740858/>
- [3] Chen et al, 2016, Advantages of Using Microwave Satellite Soil Moisture over Gridded Precipitation Products and Land Surface Model Output in Assessing Regional Vegetation Water Availability and Growth Dynamics for a Lateral Inflow Receiving Landscape. *Remote sensing* 2016, 8, 428.
- [4] Official GIMMS AVHRR download and information page. <https://nex.nasa.gov/nex/projects/1349/>
- [5] Official National Oceanic and Atmospheric Administration (NOAA) AVHRR information page. <http://noaasis.noaa.gov/NOAASIS/ml/avhrr.html>
- [6] Tucker et al, 2005, An extended AVHRR 8-km NDVI dataset compatible with MODIS and SPOT vegetation NDVI data. *International Journal of Remote Sensing*, 26.
- [7] Pinzon, J.; Tucker, C. A Non-Stationary 19812012 AVHRR NDVI3g Time Series. *Remote Sens.* 2014, 6(8), 6929-6960; doi:10.3390/rs6086929. <http://www.mdpi.com/2072-4292/6/8/6929>
- [8] Liu, Y. Y., W. A. Dorigo, et al, 2012, Trend-preserving blending of passive and active microwave soil moisture retrievals. *Remote Sensing of Environment* 123: 280-297.
- [9] Liu, Y. Y., Parinussa, R. M., Dorigo, W. A., De Jeu, R. A. M., Wagner, W., van Dijk, A. I. J. M., McCabe, M. F., Evans, J. P. , 2011, Developing an improved soil moisture dataset by blending passive and active microwave satellite-based retrievals. *Hydrology and Earth System Sciences*, 15, 425-436
- [10] Wagner, W., W. Dorigo, R. de Jeu, D. Fernandez, J. Benveniste, E. Haas, M. Ertl, 2012, *Fusion of active and passive microwave observations to create an Essential Climate Variable data record on soil moisture*. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS Annals), Volume I-7, XXII ISPRS Congress, Melbourne, Australia, 25 August-1 September 2012, 315-321
- [11] Vahid Naeimi and Wolfgang Wagner, 2010, C-band Scatterometers and Their Applications. *Geoscience and Remote Sensing New Achievements* , Chapter 13, ISBN 978-953-7619-97-8 .
- [12] Dorigo et al, 2015, Evaluation of the ESA CCI soil moisture product using ground-based observations. *Remote Sensing of Environment* 162, 380-395.
- [13] Global Land Cover 2000 database. European Commission, Joint Research Centre, 2003, <http://www-gem.jrc.it/glc2000>
- [14] Mayaux, p. et Al, The land cover of Africa for the year 2000. http://forobs.jrc.ec.europa.eu/products/glc2000/publications/workshop/march2003/Mayaux_Africa_overview.pps

8 Appendix

Code used to prepare the NDVI dataset:

```
#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python

#This code requires the following input YYYY MMM (first 3 days)to create and save NDVI data.
#This data is the combined of the 2 16 day data cycles per month NDVI provides. We average this since our
#data for soilmoisture shows better quality if we average it. This code also saves both 16 day sets of data in

import struct
import numpy as np
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt
import sys

yearinput = sys.argv[1]
monthinput = sys.argv[2]

def bin_ndarray(ndarray, new_shape, operation='sum'):
    operation = operation.lower()
    if not operation in ['sum', 'mean']:
        raise ValueError("Operation not supported.")
    if ndarray.ndim != len(new_shape):
        raise ValueError("Shape mismatch: {} -> {}".format(ndarray.shape,
                                                             new_shape))

    compression_pairs = [(d, c//d) for d,c in zip(new_shape,
                                                  ndarray.shape)]

    flattened = [l for p in compression_pairs for l in p]
    ndarray = ndarray.reshape(flattened)
    for i in range(len(new_shape)):
        op = getattr(ndarray, operation)
        ndarray = op(-1*(i+1))
print ndarray.shape
    return ndarray

datafiles=[]
folder = ''
def averaging(year,month):
    if(1981<=year<=1984):
        format='.n07-VI3g'
        folder = '1980s_new'
    if(1985<=year<=1988):
        format='.n09-VI3g'
        folder = '1980s_new'
    if(1989<=year<=1994):
        folder = '1990s_new'
        format='.n11-VI3g'
    print 'works!'
    print folder
    if(year==1989):
        folder = '1980s_new'
    if(year==1994):
```

```

if(month=='sep' or month=='oct' or month=='nov' or month=='dec'):
format='.n09-VI3g'
if(1995<=year<=2000):
folder = '1990s_new'
format='.n14-VI3g'
if(year==2000):
folder='2000s_new'
if(month=='nov' or month=='dec'):
format='.n16-VI3g'
if(2001<=year<=2003):
format='.n16-VI3g'
folder = '2000s_new'
if(2004<=year<=2008):
folder = '2000s_new'
format='.n17-VI3g'
if(2009<=year<=2011):
folder = '2010s_new'
format='.n18-VI3g'
if(year==2009):
folder='2000s_new'
if(year==2012):
format='.n19-VI3g'
folder = '2012s_new'
if(year==2013):
format='.n19-VI3g'
folder = '2013s_new'

yeartext = str(year)
yeartext = yeartext[2:]
monthtext = str(monthinput)
print folder
if(year == 1995 and month == 'jan'):
fida=open(folder + '/geo' + yeartext + monthtext + '15a.n09-VI3g',mode='rb')
else:
fida=open(folder + '/geo' + yeartext + monthtext + '15a' + format,mode='rb')

fidb=open(folder + '/geo' + yeartext + monthtext + '15b' + format,mode='rb')
fileContentA=fida.read()
fileContentB=fidb.read()

fmt=">"+"h"*(len(fileContentA)/2)
dlistA=struct.unpack(fmt,fileContentA)
dlistB=struct.unpack(fmt,fileContentB)
dataA=np.array(dlistA)
dataA=np.resize(dataA,(4320,2160))
dataA=np.transpose(dataA)
dataB=np.array(dlistB)
dataB=np.resize(dataB,(4320,2160))
dataB=np.transpose(dataB)
return dataA,dataB

dataA, dataB = averaging(int(yearinput),monthinput)

```

```

# translate in quality flag and ndvi
flagWA = dataA-np.floor(dataA/10.)*10 + 1
ndviA = np.floor(dataA/10.)/1000
flagWB = dataB-np.floor(dataB/10.)*10 + 1
ndviB = np.floor(dataB/10.)/1000

ndviReshapeA = bin_ndarray(ndviA, new_shape=(720,1440), operation='mean')
print ndviReshapeA[300,700]
ndviReshapeB = bin_ndarray(ndviB, new_shape=(720,1440), operation='mean')
print ndviReshapeB[300,700]
datafiles.append(ndviReshapeA)
datafiles.append(ndviReshapeB)

ndviMean = np.mean([datafiles[j] for j in range(len(datafiles))],axis=0)
print ndviMean[300][700]

path = "combinedNDVI/"
outfile = path+'NDVI'+yearinput+monthinput+'.npz'
np.savez(outfile,combined = ndviMean,partA=ndviReshapeA,partB=ndviReshapeB)

# plot map
lon=np.arange(1440)*0.25-180 + 1/4
lat=90.-np.arange(720)*0.25 -1/4
lons, lats = np.meshgrid(lon,lat)

fig = plt.figure()
ax = fig.add_axes([0.05,0.05,0.9,0.9])
m = Basemap(projection='kav7',lon_0=0,resolution=None)
m.drawmapboundary(fill_color='0.3')
im1 = m.pcolormesh(lons,lats,ndviMean,shading='flat',cmap=plt.cm.jet,latlon=True)
m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))
cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
cb.set_label('NDVI')
ax.set_title('AVHRR NDVI '+yearinput+' '+monthinput)
plt.show()

```

Code used to prepare the soil moisture data:

```

#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python

#This code averages the soil moisture data over 16 days.
# The code requires an YYYY MM DD input, where the DD is the last day of the 16 day dataset being created.

import struct
import numpy as np
import numpy.ma as ma
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt
from netCDF4 import Dataset
import sys

```

```

yearinput = sys.argv[1]
monthinput = sys.argv[2]
enddayinput= sys.argv[3]
#Input

yeartext = str(yearinput)
monthtext=str(monthinput)
day = str(0)+str(1)

print yearinput

my_nc_test = 'cciCombined/combined/'+yeartext+'/ESACCI-SOILMOISTURE-L3S-SSMV-COMBINED-'+ yeartext + monthtext +
print my_nc_test
test = Dataset(my_nc_test, mode='r')

datafiles=[]
def avaraging(year,month,endday):
endday = int(endday)
#opens the files, adds them to datafiles
for k in range(16):
day = endday-k
daytext = str(day)
if(day<10):
daytext=str(0)+str(day)

my_nc_file = 'cciCombined/combined/'+ yeartext + '/ESACCI-SOILMOISTURE-L3S-SSMV-COMBINED-'+ yeartext + monthtext
data = Dataset(my_nc_file, mode='r')
datafiles.append(data)
print day
return(datafiles)

avaraging(yearinput,monthinput,enddayinput)

List_of_matrices = [np.array([]) for i in range(16)]
List_of_uncertainties = [np.array([]) for i in range(16)]
print len(datafiles)
print len(List_of_matrices)

for i in range(16):
sm = datafiles[i].variables['sm'][: ]
sm = np.squeeze(sm)
List_of_matrices[i]=sm
sm_unc = datafiles[i].variables['sm_uncertainty'][: ]
sm_unc = np.squeeze(sm_unc)
List_of_uncertainties[i]=sm_unc

#avarages the datafiles to get averaged 16 day soil moisture data
sm = np.ma.mean([List_of_matrices[j] for j in range(16)], axis = 0)
sm_unc = np.ma.mean([List_of_uncertainties[j] for j in range(16)], axis = 0)

fh=datafiles[1]
lon = fh.variables['lon'][: ]
lat = fh.variables['lat'][: ]

```

```

path = 'averagedsm/16days/'
outfile1 = path+'sm16days'+ yeartext + monthtext + str(enddayinput) + '.npz'
outfile2 = path+'sm_unc16days'+ yeartext + monthtext + str(enddayinput) + '.npz'
np.ma.dump(sm,outfile1)
np.ma.dump(sm_unc,outfile2)
#saves the 16 day averaged data

fh.close()

lon_0 = lon.mean()
lat_0 = lat.mean()

lons, lats = np.meshgrid(lon,lat)

#plots the data
fig = plt.figure()
ax = fig.add_axes([0.05,0.05,0.9,0.9])
m = Basemap(projection='robin',lon_0=lon_0,lat_0=lat_0,resolution=None)
m.drawmapboundary(fill_color='0.3')
im1 = m.pcolormesh(lons,lats,np.squeeze(sm),shading='flat',cmap=plt.cm.jet,latlon=True)

m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))
cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
ax.set_title('ESA CCI soil moisture')
plt.show()

```

Code used to create the 10 year NDVI average:

```

#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python
#This code require month text input of format ttt e.g. aug
#Code creates a file containing the 10 year average of that month. This 10 year average is used as a baseline t

import struct
import numpy as np
import numpy.ma as ma
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt
from netCDF4 import Dataset
import sys

monthinput = sys.argv[1]
monthtext=str(monthinput)

print monthtext

#loads all files
ndvi1 = 'NDVI/combinedNDVI/NDVI2002'+monthtext+'.npz'
ndvi2 = 'NDVI/combinedNDVI/NDVI2003'+monthtext+'.npz'
ndvi3 = 'NDVI/combinedNDVI/NDVI2004'+monthtext+'.npz'
ndvi4 = 'NDVI/combinedNDVI/NDVI2005'+monthtext+'.npz'

```

```

ndvi5 = 'NDVI/combinedNDVI/NDVI2006'+monthtext+'.npz'
ndvi6 = 'NDVI/combinedNDVI/NDVI2007'+monthtext+'.npz'
ndvi7 = 'NDVI/combinedNDVI/NDVI2008'+monthtext+'.npz'
ndvi8 = 'NDVI/combinedNDVI/NDVI2009'+monthtext+'.npz'
ndvi9 = 'NDVI/combinedNDVI/NDVI2010'+monthtext+'.npz'
ndvi10 = 'NDVI/combinedNDVI/NDVI2011'+monthtext+'.npz'
ndvi11 = 'NDVI/combinedNDVI/NDVI2012'+monthtext+'.npz'

```

```

#loads all the 16 day subsets of the data

```

```

ndvi1 = np.load(ndvi1)
ndvi1A = ndvi1['partA']
ndvi1B = ndvi1['partB']
ndvi2 = np.load(ndvi2)
ndvi2A = ndvi2['partA']
ndvi2B = ndvi2['partB']
ndvi3 = np.load(ndvi3)
ndvi3A = ndvi3['partA']
ndvi3B = ndvi3['partB']
ndvi4 = np.load(ndvi4)
ndvi4A = ndvi4['partA']
ndvi4B = ndvi4['partB']
ndvi5 = np.load(ndvi5)
ndvi5A = ndvi5['partA']
ndvi5B = ndvi5['partB']
ndvi6 = np.load(ndvi6)
ndvi6A = ndvi6['partA']
ndvi6B = ndvi6['partB']
ndvi7 = np.load(ndvi7)
ndvi7A = ndvi7['partA']
ndvi7B = ndvi7['partB']
ndvi8 = np.load(ndvi8)
ndvi8A = ndvi8['partA']
ndvi8B = ndvi8['partB']
ndvi9 = np.load(ndvi9)
ndvi9A = ndvi9['partA']
ndvi9B = ndvi9['partB']
ndvi10 = np.load(ndvi10)
ndvi10A = ndvi10['partA']
ndvi10B = ndvi10['partB']
ndvi11 = np.load(ndvi11)
ndvi11A = ndvi11['partA']
ndvi11B = ndvi11['partB']

```

```

datafilesA=[ndvi1A,ndvi2A,ndvi3A,ndvi4A,ndvi5A,ndvi6A,ndvi7A,ndvi8A,ndvi9A,ndvi10A,ndvi11A]
datafilesB=[ndvi1B,ndvi2B,ndvi3B,ndvi4B,ndvi5B,ndvi6B,ndvi7B,ndvi8B,ndvi9B,ndvi10B,ndvi11B]

```

```

#averages the 10 year data

```

```

NDVIA = np.ma.mean([datafilesA[j] for j in range(11)], axis = 0)
NDVIB = np.ma.mean([datafilesB[j] for j in range(11)], axis = 0)

```

```

#saves the 10 year average
path = 'NDVI/10yearaverage/'
outfile1 = path+'ndvi16days'+monthtext+'averageA.npz'
outfile2 = path+'ndvi16days'+monthtext+'averageB.npz'

np.ma.dump(NDVIA,outfile1)
np.ma.dump(NDVIB,outfile2)

lon=np.arange(1440)*0.25-180 + 1/4
lat=90.-np.arange(720)*0.25 -1/4
lons, lats = np.meshgrid(lon,lat)

#plots the data
fig = plt.figure()
ax = fig.add_axes([0.05,0.05,0.9,0.9])
m = Basemap(projection='robin',lon_0 = 0,resolution=None)
m.drawmapboundary(fill_color='0.3')
im1 = m.pcolormesh(lons,lats,NDVIA,shading='flat',cmap=plt.cm.jet,latlon=True)

m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))
cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
ax.set_title('ESA CCI')
plt.show()

```

Code used to create the 10 year soil moisture average:

```

#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python
#This code requires a MM A/B input. A stands for first half of the month, B for the second half of the month be
#Code creates a file containing the 10 year average of that month. This 10 year average is used as a baseline t

import struct
import numpy as np
import numpy.ma as ma
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt
from netCDF4 import Dataset
import sys

#This code averages the soilmoisturedata from the years 2002 to 2012 per month per 16 days.
#WORKS FOR ALL EXCEPT FEB. FOR FEBRUARI MANUALLY INPUT FILES
monthinput = sys.argv[1]
partinput = sys.argv[2]
monthtext=str(monthinput)
parttext = str(partinput)

#test to see if path is correct
print monthtext
inputpath = 'averagedsm/16days/'
my_nc_test = inputpath+'sm16days20020116.npz'
print my_nc_test

```

```

#translates the A/B input in format that works with the filename
if (parttext == 'A'):
    days = str(16)

if (parttext == 'B'):
if(monthtext in('01','03','05','07','08','10','12')):
    days = str(31)
else:
    days = str(30)
#loads all files
sm1=inputpath+'sm16days2002'+monthtext+days+'.npz'
sm2=inputpath+'sm16days2003'+monthtext+days+'.npz'
sm3=inputpath+'sm16days2004'+monthtext+days+'.npz'
sm4=inputpath+'sm16days2005'+monthtext+days+'.npz'
sm5=inputpath+'sm16days2006'+monthtext+days+'.npz'
sm6=inputpath+'sm16days2007'+monthtext+days+'.npz'
sm7=inputpath+'sm16days2008'+monthtext+days+'.npz'
sm8=inputpath+'sm16days2009'+monthtext+days+'.npz'
sm9=inputpath+'sm16days2010'+monthtext+days+'.npz'
sm10=inputpath+'sm16days2011'+monthtext+days+'.npz'
sm11=inputpath+'sm16days2012'+monthtext+days+'.npz'

datafiles=[np.load(sm1),np.load(sm2),np.load(sm3),np.load(sm4),np.load(sm5),np.load(sm6),np.load(sm7),np.load(sm8),np.load(sm9),np.load(sm10),np.load(sm11)]

#averages the data
sm = np.ma.mean([datafiles[j] for j in range(11)], axis = 0)

#saves the data
path = 'averagedsm/10yearaverage/'
outfile = path+'sm16daysaverage'+ monthtext +str(partinput) + '.npz'

np.ma.dump(sm,outfile)

lon=np.arange(1440)*0.25-180 + 1/4
lat=90.-np.arange(720)*0.25 -1/4
lons, lats = np.meshgrid(lon,lat)

#plots the data
fig = plt.figure()
ax = fig.add_axes([0.05,0.05,0.9,0.9])
m = Basemap(projection='robin',lon_0 = 0,resolution=None)
m.drawmapboundary(fill_color='0.3')
im1 = m.pcolormesh(lons,lats,sm,shading='flat',cmap=plt.cm.jet,latlon=True)

m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))
cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
ax.set_title('ESA CCI')
plt.show()

```

Code that calculates the correlation coefficients based on the deviations for a specific year. This is the code for the no lag result:

```
#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python
```

```
#Code that calculates the deviation from the 10 year average, and then calculates the Pearson correlation coeff
#code requires a YYYY input.
```

```
import struct
import numpy as np
import numpy.ma as ma
from mpl_toolkits.basemap import Basemap, maskoceans
import matplotlib.pyplot as plt
import sys
import time
```

```
start_time = time.time()
```

```
yearinput = sys.argv[1]
yeartext = str(yearinput)
```

```
#File containing plotting information:
```

```
lonlat = np.load('lonlat.npz')
lon = lonlat['lon']
lat = lonlat['lat']
lonlat.close()
TestcoordX = 720
TestcoordY = 310
```

```
#LOAD ALL RELEVANT FILES
```

```
smpath = 'averagedsm/16days/'
sm1A = smpath+'sm16days'+yeartext+'0116.npz'
sm1B = smpath+'sm16days'+yeartext+'0131.npz'
sm2A = smpath+'sm16days'+yeartext+'0116.npz'
if(int(yearinput)%4 == 0):
sm2B = smpath+'sm16days'+yeartext+'0229.npz'
else:
sm2B = smpath+'sm16days'+yeartext+'0228.npz'

sm3A = smpath+'sm16days'+yeartext+'0316.npz'
sm3B = smpath+'sm16days'+yeartext+'0331.npz'
sm4A = smpath+'sm16days'+yeartext+'0416.npz'
sm4B = smpath+'sm16days'+yeartext+'0430.npz'
sm5A = smpath+'sm16days'+yeartext+'0516.npz'
sm5B = smpath+'sm16days'+yeartext+'0531.npz'
sm6A = smpath+'sm16days'+yeartext+'0616.npz'
sm6B = smpath+'sm16days'+yeartext+'0630.npz'
sm7A = smpath+'sm16days'+yeartext+'0716.npz'
sm7B = smpath+'sm16days'+yeartext+'0731.npz'
sm8A = smpath+'sm16days'+yeartext+'0816.npz'
sm8B = smpath+'sm16days'+yeartext+'0831.npz'
sm9A = smpath+'sm16days'+yeartext+'0916.npz'
sm9B = smpath+'sm16days'+yeartext+'0930.npz'
sm10A = smpath+'sm16days'+yeartext+'1016.npz'
sm10B = smpath+'sm16days'+yeartext+'1031.npz'
sm11A = smpath+'sm16days'+yeartext+'1116.npz'
sm11B = smpath+'sm16days'+yeartext+'1130.npz'
```

```

sm12A = smpath+'sm16days'+yeartext+'1216.npz'
sm12B = smpath+'sm16days'+yeartext+'1231.npz'
print 'Soil Moisture loaded'

ndvi1 = 'NDVI/combinedNDVI/NDVI'+yeartext+'jan.npz'
ndvi2 = 'NDVI/combinedNDVI/NDVI'+yeartext+'feb.npz'
ndvi3 = 'NDVI/combinedNDVI/NDVI'+yeartext+'mar.npz'
ndvi4 = 'NDVI/combinedNDVI/NDVI'+yeartext+'apr.npz'
ndvi5 = 'NDVI/combinedNDVI/NDVI'+yeartext+'may.npz'
ndvi6 = 'NDVI/combinedNDVI/NDVI'+yeartext+'jun.npz'
ndvi7 = 'NDVI/combinedNDVI/NDVI'+yeartext+'jul.npz'
ndvi8 = 'NDVI/combinedNDVI/NDVI'+yeartext+'aug.npz'
ndvi9 = 'NDVI/combinedNDVI/NDVI'+yeartext+'sep.npz'
ndvi10 = 'NDVI/combinedNDVI/NDVI'+yeartext+'oct.npz'
ndvi11 = 'NDVI/combinedNDVI/NDVI'+yeartext+'nov.npz'
ndvi12 = 'NDVI/combinedNDVI/NDVI'+yeartext+'dec.npz'
print 'NDVI files loaded'

#LOAD MONTHLY AVERAGES TO CALCULATE DEVIATIONS
path = 'NDVI/10yearaverage/'
averagelistNDVIfiles = [path+'ndvi16daysjanaverageA.npz',path+'ndvi16daysjanaverageB.npz',path+'ndvi16daysfebav
averageNDVI = []
for i in range(len(averagelistNDVIfiles)):
file = np.load(averagelistNDVIfiles[i])
averageNDVI.append(file)

print 'NDVI Averages loaded'

ndvi1 = np.load(ndvi1)
ndvi1A = ndvi1['partA']
ndvi1B = ndvi1['partB']
ndvi2 = np.load(ndvi2)
ndvi2A = ndvi2['partA']
ndvi2B = ndvi2['partB']
ndvi3 = np.load(ndvi3)
ndvi3A = ndvi3['partA']
ndvi3B = ndvi3['partB']
ndvi4 = np.load(ndvi4)
ndvi4A = ndvi4['partA']
ndvi4B = ndvi4['partB']
ndvi5 = np.load(ndvi5)
ndvi5A = ndvi5['partA']
ndvi5B = ndvi5['partB']
ndvi6 = np.load(ndvi6)
ndvi6A = ndvi6['partA']
ndvi6B = ndvi6['partB']
ndvi7 = np.load(ndvi7)
ndvi7A = ndvi7['partA']
ndvi7B = ndvi7['partB']
ndvi8 = np.load(ndvi8)
ndvi8A = ndvi8['partA']

```

```

ndvi8B = ndvi8['partB']
ndvi9 = np.load(ndvi9)
ndvi9A = ndvi9['partA']
ndvi9B = ndvi9['partB']
ndvi10 = np.load(ndvi10)
ndvi10A = ndvi10['partA']
ndvi10B = ndvi10['partB']
ndvi11 = np.load(ndvi11)
ndvi11A = ndvi11['partA']
ndvi11B = ndvi11['partB']
ndvi12 = np.load(ndvi12)
ndvi12A = ndvi12['partA']
ndvi12B = ndvi12['partB']
NDVIfileList = [ndvi1,ndvi2,ndvi3,ndvi4,ndvi5,ndvi6,ndvi7,ndvi8,ndvi9,ndvi10,ndvi11,ndvi12]
for file in NDVIfileList:
file.close()

NDVIList = [ndvi1A.flatten(),ndvi1B.flatten(),ndvi2A.flatten(),ndvi2B.flatten(),ndvi3A.flatten(),ndvi3B.flatten()

deviationNDVIList=[]

for i in range(len(NDVIList)): #Calculates the deviation from the mean.
deviationNDVIList.append(NDVIList[i]-averageNDVI[i].flatten())

def column(matrix, i):
    return [row[i] for row in matrix]

sm1A = np.load(sm1A)
sm1B = np.load(sm1B)
sm2A = np.load(sm2A)
sm2B = np.load(sm2B)
sm3A = np.load(sm3A)
sm3B = np.load(sm3B)
sm4A = np.load(sm4A)
sm4B = np.load(sm4B)
sm5A = np.load(sm5A)
sm5B = np.load(sm5B)
sm6A = np.load(sm6A)
sm6B = np.load(sm6B)
sm7A = np.load(sm7A)
sm7B = np.load(sm7B)
sm8A = np.load(sm8A)
sm8B = np.load(sm8B)
sm9A = np.load(sm9A)
sm9B = np.load(sm9B)
sm10A = np.load(sm10A)
sm10B = np.load(sm10B)
sm11A = np.load(sm11A)
sm11B = np.load(sm11B)
sm12A = np.load(sm12A)
sm12B = np.load(sm12B)

path = 'averagedsm/10yearaverage/'
averageSMlist = [np.load(path+'sm16daysaverage01A.npz'),np.load(path+'sm16daysaverage01B.npz'),np.load(path+'sm

```

```

deviationSM = []
smlist = [sm1A.flatten(),sm1B.flatten(),sm2A.flatten(),sm2B.flatten(),sm3A.flatten(),sm3B.flatten(),sm4A.flatter
smfilelist = [sm1A,sm1B,sm2A,sm2B,sm3A,sm3B,sm4A,sm4B,sm5A,sm5B,sm6A,sm6B,sm7A,sm7B,sm8A,sm8B,sm9A,sm9B,sm10A,s

for i in range(len(smlist)): #Calculates the deviation from the mean.
deviationSM.append(smlist[i]-averageSmlist[i].flatten())

print len(deviationSM)

ndvidataaugust = np.reshape(deviationNDVIList[13],(720,1440))
smdataaugust = np.reshape(deviationSM[13],(720,1440))

ndvi = np.stack(deviationNDVIList)
print ndvi.shape
sm = np.stack(deviationSM)
print sm.shape
print column(ndvi,TestcoordY*1440 + TestcoordX)
print column(sm,TestcoordY*1440 + TestcoordX)
print np.ma.corrcoef(column(ndvi,TestcoordY*1440 + TestcoordX),column(sm,TestcoordY*1440 + TestcoordX))
print "Starting the correlation calculation, takes approximately 3 hours"

#imposes boundary values
NDVIBOUNDARYVALUE=0.01
SMBOUNDARYVALUE=0.01

corrResult2 = []
for i in range(len(ndvi[1,:])): #calculates the correlation coefficients. All data with less then 75% good data
if(np.ma.count_masked(column(sm,i))<0.25 * len(column(sm,i))):
if(column(ndvi,i).count(0)<0.25 * len(column(ndvi,i))):
if(np.amax(column(ndvi,i))>NDVIBOUNDARYVALUE and np.amax(column(sm,i))>SMBOUNDARYVALUE):#if deviations do not exc
corr = np.ma.corrcoef(column(ndvi,i),column(sm,i))
corr = corr[0,1]
corrResult2.append(corr)
else:
corrResult2.append(0)
else:
corrResult2.append(0)
else:
corrResult2.append(0)
end_time = time.time()
print("Elapsed time was %g seconds" % (end_time - start_time))
#print corrResult2
print len(corrResult2)
print corrResult2[TestcoordY*1440 + TestcoordX]

#for testing purposes
print np.ma.corrcoef(column(ndvi,TestcoordY*1440 + TestcoordX),column(sm,TestcoordY*1440 + TestcoordX))
corrResult2 = np.ma.masked_equal(corrResult2,999)

corrResult2 = np.reshape(corrResult2,(720,1440))

```

```

corrResult2 = np.ma.masked_invalid(corrResult2)

print corrResult2
print corrResult2[TestcoordY,TestcoordX]

#saves the result
outfile = 'deviationResults/pearsonDeviationfromMean'+yeartext+'lag0.npz'
np.ma.dump(corrResult2,outfile)
lon_0 = lon.mean()
lat_0 = lat.mean()

lons, lats = np.meshgrid(lon,lat)
ndvidataaugust = maskoceans(lons,lats,ndvidataaugust)
corrResult2 = maskoceans(lons,lats,corrResult2)
print np.ma.mean(corrResult2),'is het gemiddelde'

fig1 = plt.figure(1)
ax = fig1.add_axes([0.05,0.05,0.9,0.9])
m = Basemap(projection='robin',lon_0=lon_0,lat_0=lat_0,resolution=None)
m.drawmapboundary(fill_color='0.0')
im1 = m.pcolormesh(lons,lats,corrResult2,shading='flat',cmap=plt.cm.jet,latlon=True)

m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))
cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
cb.set_label('Pearson correlation Coefficient')

ax.set_title('Pearson Correlation of NDVI and CCI deviations, '+yeartext+'.')
plt.show()

```

Code that calculates the correlation coefficients based on the deviations for a specific year. This is the code for the 16 day lag result. The code for the other lag results is similar with the only difference being the order in which the files are loaded:

```

#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python

#Code that calculates the deviation from the 10 year average, and then calculates the Pearson correlation coeff
#code requires a YYYY input. Calculates the 16 day lag results

import struct
import numpy as np
import numpy.ma as ma
from mpl_toolkits.basemap import Basemap, maskoceans
import matplotlib.pyplot as plt
import sys
import time

start_time = time.time()
yearinput = sys.argv[1]
yeartext = str(yearinput)

#File containing plotting information:
lonlat = np.load('lonlat.npz')
lon = lonlat['lon']

```

```

lat = lonlat['lat']
##Loading a list of files
lonlat.close()
TestcoordX = 720
TestcoordY = 310

#LOAD ALL RELEVANT FILES
smpath = 'averagedsm/16days/'
sm1A = smpath+'sm16days'+yeartext+'0116.npz'
sm1B = smpath+'sm16days'+yeartext+'0131.npz'
sm2A = smpath+'sm16days'+yeartext+'0116.npz'
if(int(yearinput)%4 == 0):
sm2B = smpath+'sm16days'+yeartext+'0229.npz'
else:
sm2B = smpath+'sm16days'+yeartext+'0228.npz'
sm3A = smpath+'sm16days'+yeartext+'0316.npz'
sm3B = smpath+'sm16days'+yeartext+'0331.npz'
sm4A = smpath+'sm16days'+yeartext+'0416.npz'
sm4B = smpath+'sm16days'+yeartext+'0430.npz'
sm5A = smpath+'sm16days'+yeartext+'0516.npz'
sm5B = smpath+'sm16days'+yeartext+'0531.npz'
sm6A = smpath+'sm16days'+yeartext+'0616.npz'
sm6B = smpath+'sm16days'+yeartext+'0630.npz'
sm7A = smpath+'sm16days'+yeartext+'0716.npz'
sm7B = smpath+'sm16days'+yeartext+'0731.npz'
sm8A = smpath+'sm16days'+yeartext+'0816.npz'
sm8B = smpath+'sm16days'+yeartext+'0831.npz'
sm9A = smpath+'sm16days'+yeartext+'0916.npz'
sm9B = smpath+'sm16days'+yeartext+'0930.npz'
sm10A = smpath+'sm16days'+yeartext+'1016.npz'
sm10B = smpath+'sm16days'+yeartext+'1031.npz'
sm11A = smpath+'sm16days'+yeartext+'1116.npz'
sm11B = smpath+'sm16days'+yeartext+'1130.npz'
sm12A = smpath+'sm16days'+yeartext+'1216.npz'
sm12B = smpath+'sm16days'+yeartext+'1231.npz'
print 'Soil Moisture loaded'

ndvi1 = 'NDVI/combinedNDVI/NDVI'+yeartext+'jan.npz'
ndvi2 = 'NDVI/combinedNDVI/NDVI'+yeartext+'feb.npz'
ndvi3 = 'NDVI/combinedNDVI/NDVI'+yeartext+'mar.npz'
ndvi4 = 'NDVI/combinedNDVI/NDVI'+yeartext+'apr.npz'
ndvi5 = 'NDVI/combinedNDVI/NDVI'+yeartext+'may.npz'
ndvi6 = 'NDVI/combinedNDVI/NDVI'+yeartext+'jun.npz'
ndvi7 = 'NDVI/combinedNDVI/NDVI'+yeartext+'jul.npz'
ndvi8 = 'NDVI/combinedNDVI/NDVI'+yeartext+'aug.npz'
ndvi9 = 'NDVI/combinedNDVI/NDVI'+yeartext+'sep.npz'
ndvi10 = 'NDVI/combinedNDVI/NDVI'+yeartext+'oct.npz'
ndvi11 = 'NDVI/combinedNDVI/NDVI'+yeartext+'nov.npz'
ndvi12 = 'NDVI/combinedNDVI/NDVI'+yeartext+'dec.npz'
ndvi13 = 'NDVI/combinedNDVI/NDVI'+str(int(yearinput)+1)+'jan.npz'
print 'NDVI files loaded'

#LOAD MONTHLY AVERAGES TO CALCULATE DEVIATIONS
path = 'NDVI/10yearaverage/'

```

```

averagelistNDVIfiles = [path+'ndvi16daysjanaverageB.npz',path+'ndvi16daysfebaverageA.npz',path+'ndvi16daysfebav
averageNDVI = []
for i in range(len(averagelistNDVIfiles)):
file = np.load(averagelistNDVIfiles[i])
averageNDVI.append(file)

```

```

print 'NDVI Averages loaded'

```

```

ndvi1 = np.load(ndvi1)
ndvi1A = ndvi1['partB']
#ndvi1B = ndvi1['partB']
ndvi2 = np.load(ndvi2)
ndvi1B = ndvi2['partA']
ndvi2A = ndvi2['partB']
ndvi3 = np.load(ndvi3)
ndvi2B = ndvi3['partA']
ndvi3A = ndvi3['partB']
ndvi4 = np.load(ndvi4)
ndvi3B = ndvi4['partA']
ndvi4A = ndvi4['partB']
ndvi5 = np.load(ndvi5)
ndvi4B = ndvi5['partA']
ndvi5A = ndvi5['partB']
ndvi6 = np.load(ndvi6)
ndvi5B = ndvi6['partA']
ndvi6A = ndvi6['partB']
ndvi7 = np.load(ndvi7)
ndvi6B = ndvi7['partA']
ndvi7A = ndvi7['partB']
ndvi8 = np.load(ndvi8)
ndvi7B = ndvi8['partA']
ndvi8A = ndvi8['partB']
ndvi9 = np.load(ndvi9)
ndvi8B = ndvi9['partA']
ndvi9A = ndvi9['partB']
ndvi10 = np.load(ndvi10)
ndvi9B = ndvi10['partA']
ndvi10A = ndvi10['partB']
ndvi11 = np.load(ndvi11)
ndvi10B = ndvi11['partA']
ndvi11A = ndvi11['partB']
ndvi12 = np.load(ndvi12)
ndvi11B = ndvi12['partA']
ndvi12A = ndvi12['partB']
ndvi13 = np.load(ndvi13)
ndvi12B = ndvi13['partA']
NDVIfileList = [ndvi1,ndvi2,ndvi3,ndvi4,ndvi5,ndvi6,ndvi7,ndvi8,ndvi9,ndvi10,ndvi11,ndvi12,ndvi13]
for file in NDVIfileList:
file.close()

```

```

NDVIList = [ndvi1A.flatten(),ndvi1B.flatten(),ndvi2A.flatten(),ndvi2B.flatten(),ndvi3A.flatten(),ndvi3B.flatten

```

```

deviationNDVIList=[]

for i in range(len(NDVIList)): #Calculates the deviation from the mean.
deviationNDVIList.append(NDVIList[i]-averageNDVI[i].flatten())

def column(matrix, i):
    return [row[i] for row in matrix]

sm1A = np.load(sm1A)
sm1B = np.load(sm1B)
sm2A = np.load(sm2A)
sm2B = np.load(sm2B)
sm3A = np.load(sm3A)
sm3B = np.load(sm3B)
sm4A = np.load(sm4A)
sm4B = np.load(sm4B)
sm5A = np.load(sm5A)
sm5B = np.load(sm5B)
sm6A = np.load(sm6A)
sm6B = np.load(sm6B)
sm7A = np.load(sm7A)
sm7B = np.load(sm7B)
sm8A = np.load(sm8A)
sm8B = np.load(sm8B)
sm9A = np.load(sm9A)
sm9B = np.load(sm9B)
sm10A = np.load(sm10A)
sm10B = np.load(sm10B)
sm11A = np.load(sm11A)
sm11B = np.load(sm11B)
sm12A = np.load(sm12A)
sm12B = np.load(sm12B)

path = 'averagedsm/10yearaverage/'
averageSMlist = [np.load(path+'sm16daysaverage01A.npz'), np.load(path+'sm16daysaverage01B.npz'), np.load(path+'sm16daysaverage02A.npz'), np.load(path+'sm16daysaverage02B.npz'), np.load(path+'sm16daysaverage03A.npz'), np.load(path+'sm16daysaverage03B.npz'), np.load(path+'sm16daysaverage04A.npz'), np.load(path+'sm16daysaverage04B.npz'), np.load(path+'sm16daysaverage05A.npz'), np.load(path+'sm16daysaverage05B.npz'), np.load(path+'sm16daysaverage06A.npz'), np.load(path+'sm16daysaverage06B.npz'), np.load(path+'sm16daysaverage07A.npz'), np.load(path+'sm16daysaverage07B.npz'), np.load(path+'sm16daysaverage08A.npz'), np.load(path+'sm16daysaverage08B.npz'), np.load(path+'sm16daysaverage09A.npz'), np.load(path+'sm16daysaverage09B.npz'), np.load(path+'sm16daysaverage10A.npz'), np.load(path+'sm16daysaverage10B.npz')]
deviationSM = []
smlist = [sm1A.flatten(), sm1B.flatten(), sm2A.flatten(), sm2B.flatten(), sm3A.flatten(), sm3B.flatten(), sm4A.flatten(), sm4B.flatten(), sm5A.flatten(), sm5B.flatten(), sm6A.flatten(), sm6B.flatten(), sm7A.flatten(), sm7B.flatten(), sm8A.flatten(), sm8B.flatten(), sm9A.flatten(), sm9B.flatten(), sm10A.flatten(), sm10B.flatten(), sm11A.flatten(), sm11B.flatten(), sm12A.flatten(), sm12B.flatten()]
smfilelist = [sm1A, sm1B, sm2A, sm2B, sm3A, sm3B, sm4A, sm4B, sm5A, sm5B, sm6A, sm6B, sm7A, sm7B, sm8A, sm8B, sm9A, sm9B, sm10A, sm10B, sm11A, sm11B, sm12A, sm12B]

for i in range(len(smlist)): #Calculates the deviation from the mean.
deviationSM.append(smlist[i]-averageSMlist[i].flatten())

print len(deviationSM)

ndvidataaugust = np.reshape(deviationNDVIList[13], (720, 1440))
smdataaugust = np.reshape(deviationSM[13], (720, 1440))

```

```

ndvi = np.stack(deviationNDVIList)
print ndvi.shape
sm = np.stack(deviationSM)
print sm.shape
print column(ndvi,TestcoordY*1440 + TestcoordX)
print column(sm,TestcoordY*1440 + TestcoordX)
print np.ma.corrcoef(column(ndvi,TestcoordY*1440 + TestcoordX),column(sm,TestcoordY*1440 + TestcoordX))
print "Starting the loop of duurt lang"

#imposes boundary values

NDVIBOUNDARYVALUE=0.01
SMBOUNDARYVALUE=0.01

corrResult2 = []
for i in range(len(ndvi[1,:])):
if(np.ma.count_masked(column(sm,i))<0.25 * len(column(sm,i))): #calculates the correlation coefficients. All da
if(column(ndvi,i).count(0)<0.25 * len(column(ndvi,i))):
if(np.amax(column(ndvi,i))>NDVIBOUNDARYVALUE and np.amax(column(sm,i))>SMBOUNDARYVALUE):#if deviations do not exc
corr = np.ma.corrcoef(column(ndvi,i),column(sm,i))
corr = corr[0,1]
corrResult2.append(corr)
else:
corrResult2.append(0)
else:
corrResult2.append(0)
else:
corrResult2.append(0)
end_time = time.time()
print("Elapsed time was %g seconds" % (end_time - start_time))
#print corrResult2
print len(corrResult2)
print corrResult2[TestcoordY*1440 + TestcoordX]

print np.ma.corrcoef(column(ndvi,TestcoordY*1440 + TestcoordX),column(sm,TestcoordY*1440 + TestcoordX))

corrResult2 = np.reshape(corrResult2,(720,1440))
corrResult2 = np.ma.masked_invalid(corrResult2)

#for testing purposes
print corrResult2
print corrResult2[TestcoordY,TestcoordX]

#print np.ma.max(corrResult2)
#print np.ma.min(corrResult2)

#saves the result
outfile = 'deviationResults/pearsonDeviationfromMean'+yeartext+'lag1.npz'

np.ma.dump(corrResult2,outfile)
lon_0 = lon.mean()
lat_0 = lat.mean()
#plots the result
lons, lats = np.meshgrid(lon,lat)

```

```

ndvidataaugust = maskoceans(lons,lats,ndvidataaugust)
corrResult2 = maskoceans(lons,lats,corrResult2)
print np.ma.mean(corrResult2), 'is het gemiddelde'

fig1 = plt.figure(1)
ax = fig1.add_axes([0.05,0.05,0.9,0.9])
m = Basemap(projection='robin',lon_0=lon_0,lat_0=lat_0,resolution=None)
#m.drawcoastlines()
#m.drawcountries()
m.drawmapboundary(fill_color='0.0')
im1 = m.pcolormesh(lons,lats,corrResult2,shading='flat',cmap=plt.cm.jet,latlon=True)

m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))
cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
cb.set_label('Pearson correlation Coefficient')

ax.set_title('Pearson Correlation of NDVI and CCI deviations, '+yeartext+'. 16 day lag.')
plt.show()

```

Code used to transform the GLC map:

```

#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python

#Code that rescales landcover to a smaller resolution
#If 75% or more of pixels that fall in the new smaller resolution the new value will become that of the 75%
#The standard landcover dataset comes with 19 different data values that are discrete values that describe the

import struct
import numpy as np
import numpy.ma as ma
from mpl_toolkits.basemap import Basemap, maskoceans
import matplotlib.pyplot as plt
import sys
import time

start_time = time.time()
#loads the data
nlon = 40320
nlat = 16353
fid=open('glc2000_v1_1.bil',mode='rb')
fileContent=fid.read()
dtype = np.uint8
print len(fileContent)

dlist=np.fromfile('glc2000_v1_1.bil',dtype=dtype)
print len(dlist)

data=np.resize(dlist,(nlat,nlon))
print data.shape

##Adding water for all the data points with lat from -56 to -90

```

```

##Since water bodies are labeled with "20" we add rows of 20 to our data to complete our set. This doesn't corr

addrows = np.ones((3807,40320))
addrows = 20*addrows

print addrows.shape
data = np.vstack([data,addrows])

print data.shape

ResolutionXnew=1440
ResolutionYnew=720

ResolutionX=nlon # RESOLUTION X GLC HERE WORKS IF RESOLUTIONX AND Y ARE DEVIDABLE BY 720
ResolutionY=nlat + 3807 # RESOLUTION Y GLC HERE

GLCnewRes=np.zeros((ResolutionYnew,ResolutionXnew),dtype=np.float)
print GLCnewRes.shape

#define the scaling coefficients
xscale = ResolutionX/ResolutionXnew
yscale = ResolutionY/ResolutionYnew
rescaledpoints = xscale*yscale
minPercent = 0.75 #minPercent indicates the 75 percentage rule

pixelList=[]
legendaList=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23] #numbers that are represent a type of
newLegenda=[1,8,9,10,12,18,19,20,21,23,24] #1 stands for lot of tree coverage.10 is used for burnt trees and re
#18 is used for cultivated regions / croplands. 20 remains unchanged (still bare areas). 21 is used for water b
colorlist = [(0, .39,0,1),(0, .5843,0,1),(0.6823,0.9960,0.3843,1),(0.5411,0.266667,0.070588,1),(0.8,0.49411,0.372
print len(colorlist)
#List of rgb alpha values to match to official GLC 2000 legend.

#imposes the new defined ecosystem classes
data[data <7] = 1
data[data == 7] = 8
data[data == 11] = 12
data[data == 13] = 12
data[data == 14] = 12
data[data == 15] = 8
data[data == 16] = 18
data[data == 17] = 18
data[data == 21] = 20
data[data == 22] = 23

#code that does the actual transforming
for i in range(ResolutionYnew):
for j in range(ResolutionXnew):
for k in range(i*xscale, (i+1)*xscale):
for l in range(j*yscale, (j+1)*yscale):
pixelList.append(data[k,l])
for type in newLegenda:

```

```

if(pixelList.count(type)>= minPercent * len(pixelList)): #If there is more then 75% of one type change it to th
GLCnewRes[i,j]=type
break
else:
GLCnewRes[i,j]=24

pixelList = []

end_time = time.time()
print("Elapsed time was %g seconds" % (end_time - start_time))

#saves the map
outfile = 'rescaledMap.npz'
np.ma.dump(GLCnewRes,outfile)

#PLOT GLCnewRES
lon=np.arange(1440)*0.25-180 + 1/4
lat=90.-np.arange(720)*0.25 -1/4
lons, lats = np.meshgrid(lon,lat)

#Creating custom discrete color map
newLegenda=[1,8,9,10,12,18,19,20,23,24]
#We transform the legend to 1,2,3 etc for the purpose of color only. so 1 = 1, 2 = 8, 3 = 9 etc.
GLCnewRes [GLCnewRes==8]=2
GLCnewRes [GLCnewRes==9]=3
GLCnewRes [GLCnewRes==10]=4
GLCnewRes [GLCnewRes==12]=5
GLCnewRes [GLCnewRes==18]=6
GLCnewRes [GLCnewRes==19]=7
GLCnewRes [GLCnewRes==20]=8
GLCnewRes [GLCnewRes==23]=9
GLCnewRes [GLCnewRes==24]=10

#defines colors according to the original colormap
newcolorlist=[(0,.39,0,1),(0,0.2745,0.78039,1),(0,0.898,0,1),(0,0,0,1),(0.99607,0.6980,0,1),(0.78823,0.5375,0.9

#creating a custom color map
cmap = plt.get_cmap('jet', 10)
cmaplist = [cmap(i) for i in range(cmap.N)]
for i in range(len(newcolorlist)-1):
cmaplist[i]=newcolorlist[i]
print len(cmaplist)
cmap = cmap.from_list('Custom cmap', cmaplist, cmap.N)

#plots the actual figure
fig1 = plt.figure(1)
ax = fig1.add_axes([0.05,0.05,0.9,0.9])
m = Basemap(projection='robin',lon_0=0,resolution=None)
m.drawmapboundary(fill_color='0.0')
im1 = m.pcolormesh(lons,lats,GLCnewRes,shading='flat',cmap=cmap,latlon=True)

m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))

```

```
#cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
ax.set_title('landcover map at 0.25 degree resolution')
```

```
plt.show()
```

Code used to create the average pearson correlation for 10 years from the results:

```
#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python

#Code that calculates the yearly averaged soil moisture and NDVI deviations.
#This code takes the monthly deviatons in sm and ndvi and averages them over a year.
#This code will show the deviations both absolute, and relative to the region.

import struct
import numpy as np
import numpy.ma as ma
from mpl_toolkits.basemap import Basemap, maskoceans
import matplotlib.pyplot as plt
import sys
import time

lag = sys.argv[1]
pearsonlist=[]

if lag == '0':
lagtext = 'No lag'
elif lag == '1':
lagtext = '16 day lag'
elif lag == '2':
lagtext = '32 day lag'
elif lag == '-1':
lagtext = '-16 day lag'

for i in range(1,12):
yeartext = str(2001+i)
file = np.load('deviationResults/pearsonDeviationfromMean'+yeartext+'lag'+str(lag)+'.npz')
print yeartext
pearsonlist.append(file)

#Import all the monthly data files for the requested year

start_time = time.time()

Pearsonyear = np.mean([pearsonlist[j] for j in range(len(pearsonlist))], axis = 0)

#REMOVE NAN DUE TO EARLIER MASKING
lon=np.arange(1440)*0.25-180 + 1/4
lat=90.-np.arange(720)*0.25 -1/4
lons, lats = np.meshgrid(lon,lat)

Pearsonyear = maskoceans(lons,lats,Pearsonyear)

fig1 = plt.figure(1)
ax = fig1.add_axes([0.05,0.05,0.9,0.9])
```

```

m = Basemap(projection='robin',lon_0=0,resolution=None)
m.drawmapboundary(fill_color='0.0')
im1 = m.pcolormesh(lons,lats,Pearsonyear,shading='flat',cmap=plt.cm.jet,latlon=True)
m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))
cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
cb.set_label('Pearson correlation coefficient')
ax.set_title('Average Pearson correlation in the 2002-2012 period. '+lagtext)

plt.show()

```

Code used to link the GLC data set to correlation results. Code for the comparison to the GLC data, without the comparison of the different lag types differences only in skipping the difference calculation.

```
#!/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin/python
```

```
#Code that shows the ecosystems of the requested lag difference that cross the boundry value
```

```
#Requires a boundry for significant coefficients. Requires a BB LA LB input. BB is a significance boundry, LA is
```

```

import struct
import numpy as np
import numpy.ma as ma
from mpl_toolkits.basemap import Basemap, maskoceans
import matplotlib.pyplot as plt
import sys
import time

```

```

correlationboundry=sys.argv[1]
lagA = sys.argv[2]
lagB = sys.argv[3]

```

```

if lagA=='-1':
lagA=Min1

```

```

if lagB=='-1':
lagB='Min1'

```

```

pearsonlistlagMin1=[]
pearsonlist0=[]
pearsonlist1=[]
pearsonlist2=[]

```

```

#loads the files
for i in range(1,12):
yeartext = str(2001+i)
file = np.load('deviationResults/pearsonDeviationfromMean'+yeartext+'lag-1.npz')
print yeartext
pearsonlistlagMin1.append(file)
for i in range(1,12):
yeartext = str(2001+i)
file = np.load('deviationResults/pearsonDeviationfromMean'+yeartext+'lag0.npz')
print yeartext
pearsonlist0.append(file)
for i in range(1,12):

```

```

yeartext = str(2001+i)
file = np.load('deviationResults/pearsonDeviationfromMean'+yeartext+'lag1.npz')
print yeartext
pearsonlist1.append(file)
for i in range(1,12):
yeartext = str(2001+i)
file = np.load('deviationResults/pearsonDeviationfromMean'+yeartext+'lag2.npz')
print yeartext
pearsonlist2.append(file)

#Import all the monthly data files for the requested year

start_time = time.time()
#calculates the mean
PearsonyearMin1 = np.mean([pearsonlistlagMin1[j] for j in range(len(pearsonlistlagMin1))], axis = 0)
Pearsonyear0 = np.mean([pearsonlist0[j] for j in range(len(pearsonlistlagMin1))], axis = 0)
Pearsonyear1 = np.mean([pearsonlist1[j] for j in range(len(pearsonlistlagMin1))], axis = 0)
Pearsonyear2 = np.mean([pearsonlist2[j] for j in range(len(pearsonlistlagMin1))], axis = 0)

#calculates the difference
Pearsonyear2min1=Pearsonyear2-Pearsonyear1
Pearsonyear2min0=Pearsonyear2-Pearsonyear0
Pearsonyear2minMin1=Pearsonyear2-PearsonyearMin1
Pearsonyear1min0=Pearsonyear1-Pearsonyear0
Pearsonyear1minMin1=Pearsonyear1-PearsonyearMin1
Pearsonyear0minMin1=Pearsonyear0-PearsonyearMin1

mydict={}
mydict['2min1']=Pearsonyear2min1
mydict['2min0']=Pearsonyear2min0
mydict['2minMin1']=Pearsonyear2minMin1
mydict['1min0']=Pearsonyear1min0
mydict['1minMin1']=Pearsonyear1minMin1
mydict['0minMin1']=Pearsonyear0minMin1

#loads the GLC map
GLCnewRes=np.load('landcover/global/Bil/rescaledMap.npz')
x=str(lagA)+'min'+str(lagB)
data = mydict[x]

#imposes the Boundry
if float(correlationboundary)<0:
data = np.ma.masked_greater(data,float(correlationboundary))
else:
data = np.ma.masked_less(data,float(correlationboundary))

mask = np.ma.getmaskarray(data)

GLCnewRes=np.ma.masked_array(GLCnewRes,mask)

lon=np.arange(1440)*0.25-180 + 1/4
lat=90.-np.arange(720)*0.25 -1/4
lons, lats = np.meshgrid(lon,lat)

```

```

#plot the data according to the GLC standard
#Creating custom discrete color map
newLegenda=[1,8,9,10,12,18,19,20,23,24]
#We transform the legend to 1,2,3 etc for the purpose of color only. so 1 = 1, 2 = 8, 3 = 9 etc.
GLCnewRes [GLCnewRes==8]=2
GLCnewRes [GLCnewRes==9]=3
GLCnewRes [GLCnewRes==10]=4
GLCnewRes [GLCnewRes==12]=5
GLCnewRes [GLCnewRes==18]=6
GLCnewRes [GLCnewRes==19]=7
GLCnewRes [GLCnewRes==20]=8
GLCnewRes [GLCnewRes==23]=9
GLCnewRes [GLCnewRes==24]=10

newcolorlist=[(0, .39,0,1),(0,0.2745,0.78039,1),(0,0.898,0,1),(0,0,0,1),(0.99607,0.6980,0,1),(0.78823,0.5375,0.9

cmap = plt.get_cmap('jet', 10)
cmaplist = [cmap(i) for i in range(cmap.N)]
for i in range(len(newcolorlist)-1):
cmaplist[i]=newcolorlist[i]
print len(cmaplist)
cmap = cmap.from_list('Custom cmap', cmaplist, cmap.N)

fig1 = plt.figure(1)
ax = fig1.add_axes([0.05,0.05,0.9,0.9])
m = Basemap(projection='robin',lon_0=0,resolution=None)
m.drawmapboundary(fill_color='1.0')
im1 = m.pcolormesh(lons,lats,GLCnewRes,shading='flat',cmap=cmap,latlon=True)
m.drawparallels(np.arange(-90.,99.,30.))
m.drawmeridians(np.arange(-180.,180.,60.))
#cb = m.colorbar(im1,"bottom", size="5%", pad="2%")
ax.set_title('landcover map at 0.25 degree resolution')

plt.show()

```