# Supplementary Material

## 1 SUPPLEMENTARY TABLES

**Table S1.** Time series available at the RegESDL. BESS: Breathing Earth System Simulator. CHIRPS: Climate Hazards group Infrared Precipitation with Stations. MODIS: Moderate Resolution Imaging Spectroradiometer. TRMM: Tropical Rainfall Measuring Mission. QA4ECV: Quality Assurance for Essential Climate Variables project. Ori.res: Original resolution.

| Variable | Abbreviation | Data product or project | Timestep* | Firstyear | Lastyear | Orig.res. |
|---|---|---|---|---|---|---|
| Bihemispherical reflectance near infrared | BHR_NIR | QA4ECV | daily | 2001 | 2018 | 0.5° |
| Bihemispherical reflectance shortwave | BHR_SW | QA4ECV | daily | 2001 | 2018 | 0.5° |
| Bihemispherical reflectance visible | BHR_VIS | QA4ECV | daily | 2001 | 2018 | 0.5° |
| Directional hemispherical reflectance near infrared | DHR_NIR | QA4ECV | daily | 2001 | 2018 | 0.5° |
| Directional hemispherical reflectance shortwave | DHR_SW | QA4ECV | daily | 2001 | 2018 | 0.5° |
| Directional hemispherical reflectance visible | DHR_VIS | QA4ECV | daily | 2011 | 2018 | 0.5° |
| Diffuse Photosynthetically Active Radiation | BESS_PARdiff_Daily | BESS | daily | 2001 | 2016 | 0.5° |
| BESS_RSDN_Daily | BESS_RSDN_Daily | BESS | daily | 2001 | 2016 | 0.5° |
| Gross Primary Productivity | GPP | BESS | 8day | 2000 | 2015 | 0.0083° |
| Photosynthetically active radiation | BESS_PAR | BESS | daily | 2001 | 2016 | 0.5° |
| Evapotranspiration | ET | BESS | 8day | 2001 | 2015 | 0.0083° |
| Enhanced Vegetation Index Aqua | EVI_aqua | MYD13A2.EVI | 16day | 2003 | 2017 | 0.0083° |
| Enhanced Vegetation Index Terra | EVI_terra | MOD13A2.EVI | 16day | 2001 | 2017 | 0.0083° |
| Fire dates | Fire_date | MCD64A1 | monthly | 2001 | 2017 | 0.0042° |
| Fire quality | Fire_quality | MCD64A1 | monthly | 2001 | 2017 | 0.0042° |
| Fire uncertainty | Fire_date_uncertainty | MCD64A1 | monthly | 2001 | 2017 | 0.0042° |
| Fraction of Absorbed PhotosyntheticallyActive Radiation | FPAR | MOD15A2 | 8day | 2001 | 2017 | 0.0042° |
| Land Surface Temperature day | LST_day | MOD11A2 | 8day | 2001 | 2017 | 0.0083° |
| Land Surface Temperature night | LST_night | MOD11A2 | 8day | 2001 | 2017 | 0.0083° |
| Leaf Area Index | LAI | MOD15A2H | 8day | 2001 | 2017 | 0.0042° |
| Normalized Difference Vegetation Index Aqua | NDVI_aqua | MYD13A2.NDVI | 16day | 2003 | 2017 | 0.0083° |
| Normalized Difference Vegetation Index Terra | NDVI_terra | MOD13A2.NDVI | 16day | 2001 | 2017 | 0.0083° |
| Vegetation Cover Fraction Trees | VCT_tree | MOD44B | annual | 2001 | 2016 | 0.0083° |
| Vegetation Cover Fraction no vegetation | VCF_nonVeg | MOD44B | annual | 2001 | 2016 | 0.0083° |
| Vegetation Cover Fraction nonTrees | VCF_nonTreeVeg | MOD44B | annual | 2001 | 2016 | 0.0083° |
| ESA_land_cover | ESA_LC | ESA Land Cover CCI Product | annual | 2001 | 2015 | 300 m |
| Precipitation | Precipitation | TRMM | monthly | 1998 | 2012 | 0.25° |
| Precipitation | Precipitation | CHIRPS | daily | 2001 | 2018 | 0.05° |

**Table S2.** Quality flags of MODIS products, and selected flag settings. QBP: Quality band position in HDF file.

| Variable | Data product | Dataset name | QBP | Data type | Quality criteria settings |
|---|---|---|---|---|---|
| LST day | MOD11A2 | QC_Day | 2 | Binary | Bit 0 to 1 == 00 & any combination<br>Bit 0 to 1 == 01 & Bit 2 to 3 == 00 or 01 or 11 |
| LST night | MOD11A2 | QC_Night | 6 | Binary | Bit 0 to 1 == 00 & any combination<br>Bit 0 to 1 == 01 & Bit 2 to 3 == 00 or 01 or 11 |
| NDVI/EVI terra | MOD13A2 | 1 km 16 days pixel reliability | 12 | 8bit | Values of 0 or 1 |
| NDVI/EVI aqua | MYD13A2 | 1 km 16 days pixel reliability | 12 | 8bit | Values of 0 or 1 |
| FPAR/LAI | MOD15A2 | FparLai_QC | 3 | Binary | Bit 0 == 0 & any combination<br>Bit 0 == 1 & Bit 5 to 7 == 000 or 001 or 010 |
| Vegetation Cover Fraction Trees/ No trees/No vegetation | MOD44B | Quality | 4 | Binary | All values $<$1111111 (This is an annual layer. Flags are fixed for dates range) |
| Fire quality | MCD64A1 | Quality | 3 | Binary | Bit 1 == 1 & any combination<br>Bit 1 == 0 & Bit 5 to 7 == from 001 to 101 |

**Table S3.** Descriptive variables available at the RegESDL. EarthEnv: Global 1-km Cloud Cover. GCH-lidar: Mapping forest canopy height globally with spaceborne lidar. HWSD: Harmonized world soil database. SoilGrids: Global Gridded Soil Information.

| Variable | Source | Resolution | Units | Citation |
|---|---|---|---|---|
| canopy_height | GCH-lidar | 1 km | m | Simard, M., Pinto N., Fisher J.B. and Baccini A. 2011. Mapping forest canopy height globally with spaceborne lidar. Journal of Geophysical Research, vol. 116. |
| clouds_monthlymean_01 | EarthEnv | 1 km | % | Wilson A.M. and Jetz W. 2016. Remotely Sensed High-Resolution Global Cloud Dynamics for Predicting Ecosystem and Biodiversity Distributions. PLoS Biol 14(3): e1002415. doi:10.1371/journal.pbio.1002415 https://www.earthenv.org/cloud |
| clouds_monthlymean_02 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_03 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_04 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_05 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_06 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_07 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_08 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_09 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_10 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_11 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| clouds_monthlymean_12 | EarthEnv | 1 km | % | Wilson and Jetz (2016) |
| available_water_storage_capacity | HWSD | 1 km | categorical_mm/m | FAO/IIASA/ISRIC/ISSCAS/JRC, 2012. Harmonized World Soil Database (version 1.2). FAO, Rome, Italy and IIASA, Laxenburg, Austria. http://webarchive.iiasa.ac.at/Research/LUC/External-World-soil-database/HTML/index.html?sb=1 |
| clay_percentage_in_subsoil | HWSD | 1 km | % | FAO et al. (2012) |
| gravel_volume_percentage_in_subsoil | HWSD | 1 km | % | FAO et al. (2012) |
| organic_carbon_weight_percentage_in_subsoil | HWSD | 1 km | % | FAO et al. (2012) |
| sand_percentage_in_subsoil | HWSD | 1 km | % | FAO et al. (2012) |
| silt_percentage_in_subsoil | HWSD | 1 km | % | FAO et al. (2012) |
| clay_percentage_in_topsoil | HWSD | 1 km | % | FAO et al. (2012) |
| gravel_volume_percentage_in_topsoil | HWSD | 1 km | % | FAO et al. (2012) |
| organic_carbon_weight_percentage_in_topsoil | HWSD | 1 km | % | FAO et al. (2012) |
| sand_percentage_in_topsoil | HWSD | 1 km | % | FAO et al. (2012) |
| silt_percentage_in_topsoil | HWSD | 1 km | % | FAO et al. (2012) |
| depth_to_bedrock_R_horizon_up_to_200_cm | SoilGrids | 250 m | cm | Hengl T, de Jesus JM, MacMillan RA, Batjes NH, Heuvelink GBM, et al. 2014. SoilGrids1km — Global Soil Information Based on Automated Mapping. PLoS ONE 9(8): e105992. doi:10.1371/journal.pone.0105992 https://www.isric.org/explore/soilgrids |
| predicted_probability_of_occurrence_of_R_horizon | SoilGrids | 250 m | % | Hengl et al. (2014) |
| absolute_depth_to_bedrock | SoilGrids | 250 m | cm | Hengl et al. (2014) |
| Soil_organic_carbon_stock | SoilGrids | 250 m | tonnes/ha | Hengl et al. (2014) |
| Soil_organic_carbon_stock | SoilGrids | 250 m | tonnes/ha | Hengl et al. (2014) |
| Bulk_density_fine_earth | SoilGrids | 250 m | kg/m3 | Hengl et al. (2014) |
| Bulk_density_fine_earth | SoilGrids | 250 m | kg/m3 | Hengl et al. (2014) |
| Clay_content_0_2_micro_meter_mass_fraction | SoilGrids | 250 m | % | Hengl et al. (2014) |
| Clay_content_0_2_micro_meter_mass_fraction | SoilGrids | 250 m | % | Hengl et al. (2014) |
| Coarse_fragments_volumetric | SoilGrids | 250 m | % | Hengl et al. (2014) |
| Coarse_fragments_volumetric | SoilGrids | 250 m | % | Hengl et al. (2014) |
| Silt_content_250_micro_meter_mass_fraction | SoilGrids | 250 m | % | Hengl et al. (2014) |
| Silt_content_2_50_micro_meter_mass_fraction | SoilGrids | 250 m | % | Hengl et al. (2014) |
| Sand_content_50–2000_micro_meter_mass_fraction | SoilGrids | 250 m | % | Hengl et al. (2014) |
| Sand_content_50–2000_micro_meter_mass_fraction | SoilGrids | 250 m | % | Hengl et al. (2014) |
| Cation_exchange_capacity_of_soil | SoilGrids | 250 m | cmolc/kg | Hengl et al. (2014) |
| Cation_exchange_capacity_of_soil | SoilGrids | 250 m | cmolc/kg | Hengl et al. (2014) |
| Soil_organic_carbon_content_fine_earth_fraction | SoilGrids | 250 m | g/kg | Hengl et al. (2014) |
| Soil_organic_carbon_content_fine_earth_fraction | SoilGrids | 250 m | g/kg | Hengl et al. (2014) |
| Soil_pH_x_10_in_H2O | SoilGrids | 250 m | pH | Hengl et al. (2014) |
| Soil_pH_x_10_in_H2O | SoilGrids | 250 m | pH | Hengl et al. (2014) |
| Soil_pH_x_10_in_KCl | SoilGrids | 250 m | pH | Hengl et al. (2014) |
| Soil_pH_x_10_in_KCl | SoilGrids | 250 m | pH | Hengl et al. (2014) |

**Table S4.** Layers of Colombia available at the RegESDL.

| Variable | Data product or project | Format-Res. | Units | Citation |
|---|---|---|---|---|
| annual_mean_precipitation _interpolation_standardized_cokriging* | Improved long-termmean annual rainfallfields for Colombia | Raster 0.017° | mm | Alvarez-Villa, O. D., Velez, J. I., & Poveda, G. 2011. Improved long-term mean annual rainfall fields for Colombia. International Journal of Climatology, 31(14), 2194–2212. https://doi.org/10.1002/joc.222 |
| annual_mean_precipitation _markov_regionalization* | Improved long-termmean annual rainfallfields for Colombia | Raster 0.017° | mm | Alvarez et al. (2011) |
| precipitation_annual_mean _interpolation_colocated_cokriging* | Improved long-termmean annual rainfallfields for Colombia | Raster 0.017° | mm | Alvarez et al. (2011) |
| precipitation_annual_mean _interpolation_kriging _with_external_drift* | Improved long-termmean annual rainfallfields for Colombia | Raster 0.017° | mm | Alvarez et al. (2011) |
| biotic_units** | Ajuste de la capa del componente biotico, incorporada dentro del Mapa Nacional de Ecosistemas Terrestres, Marinos y Costeros de Colombia, elaborado en 2016 a escala 1:100.000 | Shapefile | categorical | Londono, M. C., Bello, C., Velasquez, J., Norden, N., Ortiz, C.,Gonzalez, I., Lopez, D., Gutierrez, C., Olaya, H., and Saavedra, K.: Documento Tecnico: Componente Biotico Mapa de Ecosistemas Continentales, Marinos y Costeros de Colombia, Escala 1 : 100 000, Tech. rep., Instituto de Investigacion de Recursos Biologicos Alexander von Humboldt, Bogota, D.C., 2017http://geonetwork.humboldt.org.co/geonetwork/srv/spa/catalog.search#/metadata/a1afc35c-db98-4110-8093-98e599d1571e |
| municipalities_administrative_units*** | Cartografia Basica Digital Integrada. Republica de Colombia.. Escala 1:100.000 | Shapefile | categorical | IGAC. 2014. Cartografia Basica Digital Integrada. Republica de Colombia.. Escala 1:100.000. http://metadatos.igac.gov.co/geonetwork/srv/spa/catalog.search#/metadata/c1b4bfe5-f7c7-44a4-8849-c5f7c4257937 |
| wetlands** | Delimitacion de ecosistemas estrategicos | Raster 90 m | categorical | IAVH 2016. Identificacion de humedales, escala 1:100.000. Proyecto: Insumos para la delimitacion de ecosistemas estrategicos: Paramos y Humedales. Convenio No. 13-014 (FA. 005 de 2013) suscrito entre el Fondo Adaptacion y el Instituto Humboldt. Bogota D.C., Colombia. http://geonetwork.humboldt.org.co/geonetwork/srv/spa/catalog.search#/metadata/d68f4329-0385-47a2-8319-8b56c772b4c0 |
| agriculture_frontier*** | Identificacion General de la FronteraAgricola en Colombia a escala 1:100.000 Ministerio de Agricultura y Desarrollo Rural Agropecuario | Shapefile | categorical | Unidad de Planificacion Rural Agropecuaria (UPRA). 2017. Identificacion general de la frontera agricola en Colombia. Escala 1:100.000. MADR y UPRA. Bogota (Colombia). https://www.minagricultura.gov.co/Normatividad/Projects_Documents/IDENTIFICACION%20GENERAL%20DE%20LA%20FRONTERA%20.shp: https://sipra.upra.gov.co/ |
| national_parks*** | Parques Nacionales Naturales de Colombia | Shapefile | categorical | Limites de los Parques Nacionales Naturales de Colombia, en Sistema de Referencia Magna - Sirgas y multiescala (1:100.000 y 1:25.000). http://geonetwork.parquesnacionales.gov.co/geonetwork/srv/spa/metadata.show?id=10048&currTab=simple |

*Raster resample to RegESDL using the mean value *Categorical raster resample to RegESDL using the mode value **Categorical shapefiles raterized in R using the funcion 'last'.

**Table S5.** Data license information. BESS: Breathing Earth System Simulator. CHIRPS: Climate Hazards group Infrared Precipitation with Stations. EarthEnv: Global 1-km Cloud Cover. HWSD: Harmonized world soil database. QA4ECV: Quality Assurance for Essential Climate Variables project. MODIS: Moderate Resolution Imaging Spectroradiometer. SoilGrids: Global Gridded Soil Information. TRMM: Tropical Rainfall Measuring Mission.

| Category | Data product or project | License information |
|---|---|---|
| Time series | QA4ECV | License details in Global Attributes. Check info from ESDL |
| | BESS | License issued directly by data owner. No restriction for use. Citation to original data is mandatory. |
| | MODIS | No restrictions on subsequent use or redistributionhttps://modaps.modaps.eosdis.nasa.gov/services/faq/LAADS_Data-Use_Citation_Policies.pdf (visited on 29.07.2020) |
| | TRMM | Data are freely availablehttps://gpm.nasa.gov/data/policy |
| | CHIRPS | Creative Commons Attribution 4.0 International (CC BY 4.0) |
| Static layers | Canopy | This dataset is openly shared, without restriction, in accordance with the NASA Data and Information Policy. |
| | Harmonized World Soil Data Base | Creative Commons Attribution-Noncommercial 3.0 (CC BY-NC 3.0) |
| | SoilGrids | Creative Commons Attribution 4.0 International (CC BY 4.0) |
| | EarthEnv | Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) |
| Layers for Colombia | Improved long-term mean annual rainfall fields for Colombia | From authors |
| | biotic_units | License issued directly by data owner. No restrictiction for use. Citation to original data is mandatory. |
| | municipalities_administrative_units | Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) |
| | wetlands | TBD |
| | agriculture_frontier | Governmental information for public use without any use restrictions. Data most be properly cited. |
| | national_parks | Creative Commons Attribution-ShareAlike 3.0 Unported (CC BY-SA 3.0)License provided by Colombian National Parks Rad. 20172400004243 (http://www.parquesnacionales.gov.co/) available at: url tbf |

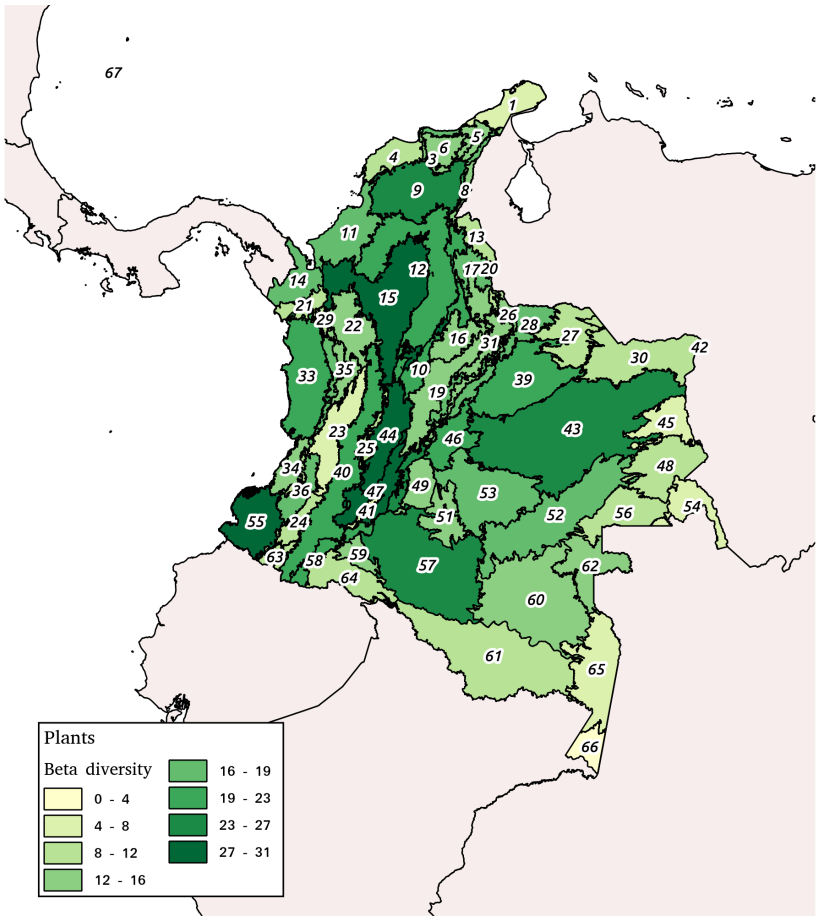
**Table S6.** Timing of computer processing when using different RegESDL version.

| Examples | Reg ESDL for Temporal analysis Time (minutes:seconds) | Reg ESDL for Spatial analysis Time (minutes:seconds) |
|---|---|---|
| Example 1: Time series decomposition using Fast Fourier | 02 min : 55 sec | 36 min : 33 sec |
| Example 2: Find the maximum value along time series | 00 min : 41 sec | 12 min : 38 sec |

**Table S7.** Scripts available in the supplementary, and at the Zenodo (`http://doi.org/10.5281/zenodo.5068004`) and GitHub (`https://github.com/linamaes/Regional_ESDL`) repositories.RegESDL: Regional Earth System Data Lab

| Scripts | Supplementary | Zenodo | GitHub |
|---|---|---|---|
| Access to the RegESDL | | | |
| 01_script_for_downloading_regional_ESDL_using_python.py | X | X | X |
| 02_loading_regional_ESDL.jl | - | X | X |
| Computing performance | | | |
| 03_comparison_TS_decomposition_using_different_DC_storage.jl | X | X | X |
| Main figures | | | |
| 01_figure_2_land_cover_map_ESA_2014.jl | - | X | X |
| 02_figure_4_principal_components_processing_&_plotting.jl | X | X | X |
| 03_figure_5_histograms_PCs_variance_explained_by_land_cover.jl | X | X | X |
| 04_figure_6_seasonality_ratio_annual_semiannual_oscillations_PC1.jl | X | X | X |
| 05_figure_7_processing_biotic_units_mean_seasonal_cycle_PC1.jl | - | X | X |
| 06_figure_7_plotting_biotic_units_mean_seasonal_cycle_PC1.jl | X | X | X |
| 07_figure_8_&_figureS4_S5_heatmaps_seasonality_ratio_&_climate.jl | X | X | X |
| Supplementary figures | | | |
| 08_figure_S2_plotting_px_MSC_4vars&pca.jl | - | X | X |
| 09_figure_S3_loadings_principal_components.jl | - | X | X |

## 2 SUPPLEMENTARY FIGURES




Biotic units

Id and Name

1. Alta Guajira
2. Estribación norte Sierra Nevada de Santa Marta
3. Estribación sur Sierra Nevada de Santa Marta
4. Cartagena y delta del Magdalena
5. Baja Guajira y alto Cesar
6. Sierra nevada de Santa Marta
7. Perijá y montes de Oca
8. Perijá
9. Ariguaní-Cesar
10. Cordillera oriental Magdalena medio
11. Sinú
12. Magdalena medio y depresión momposina
13. Zulia
14. Darién –Tacarcuna
15. Nechí-San Lucas
16. Guane-Yariguíes
17. Catatumbo
18. Tamá
19. Altoandino cordillera oriental
20. Cúcuta
21. Truandó
22. Cauca alto
23. Cauca medio
24. Patía
25. Chaparral
26. Vertiente llanera cordillera oriental
27. Arauca
28. Piedemonte Orinoquia
29. Alto Murrí
30. Bita
31. Uwa
32. Altoandino influencia llanera
33. San Juan
34. Micay
35. Estribaciones Pacífico norte
36. Estribaciones Pacífico sur
37. Vertiente Pacífico-Chocó
38. Vertiente Pacífico-Cauca
39. Casanare
40. Cordillera central
41. Caquetá influencia cordillera central
42. Maipures
43. Altillanura
44. Tolima grande
45. Matavén
46. Villavicencio
47. Huila-Caquetá
48. Guainía
49. Macarena
50. Picachos
51. Alto Guayabero
52. Inírida-Papunaua
53. Guaviare – Guayabero
54. Serranía del Naquén
55. Pacífico nariñense-Tumaco
56. Puinawai
57. Yarí-Chiribiquete
58. Piedemonte Amazonas
59. Alto Caquetá
60. Apaporis
61. Huitoto-Cahuinarí
62. Bajo Vaupés
63. Nudo de los pastos
64. Alto Putumayo
65. Bajo Caquetá - Puré
66. Ticuna – Amacayacu
67. San Andrés y Providencia

**Figure S1.** Biotic units of Colombia. Values correspond to Beta diversity values of plants (Magnoliopsida).

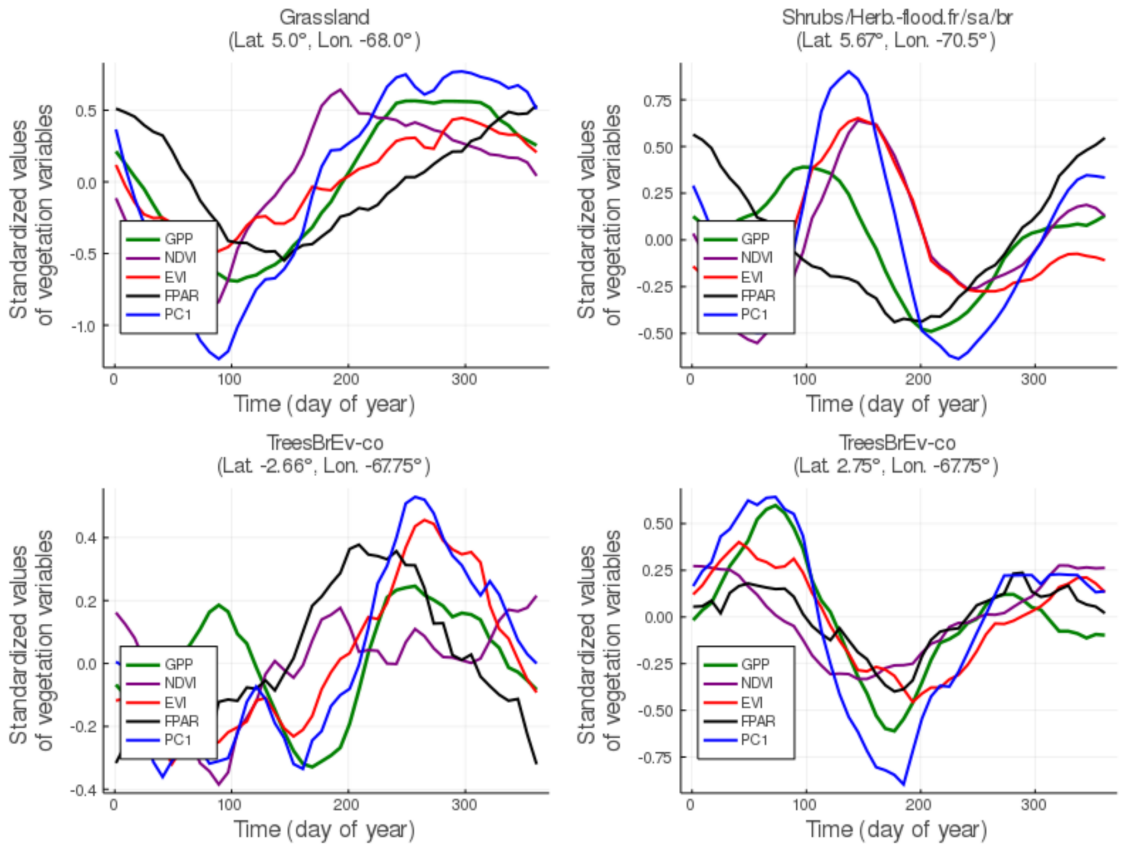



**Figure S2.** Mean seasonal cycle of the vegetation variables set calculated from the first principal component. Variables have been standardized to $\mu = 0$ and $\delta = 1$. Shrubs/Herb.-flood.fr/sa/br: Shrub or herbaceous cover, flooded, fresh/saline/brakish water. TreeBrEv-co: Tree cover, broadleaved, evergreen, closed to open ($> 15\%$).

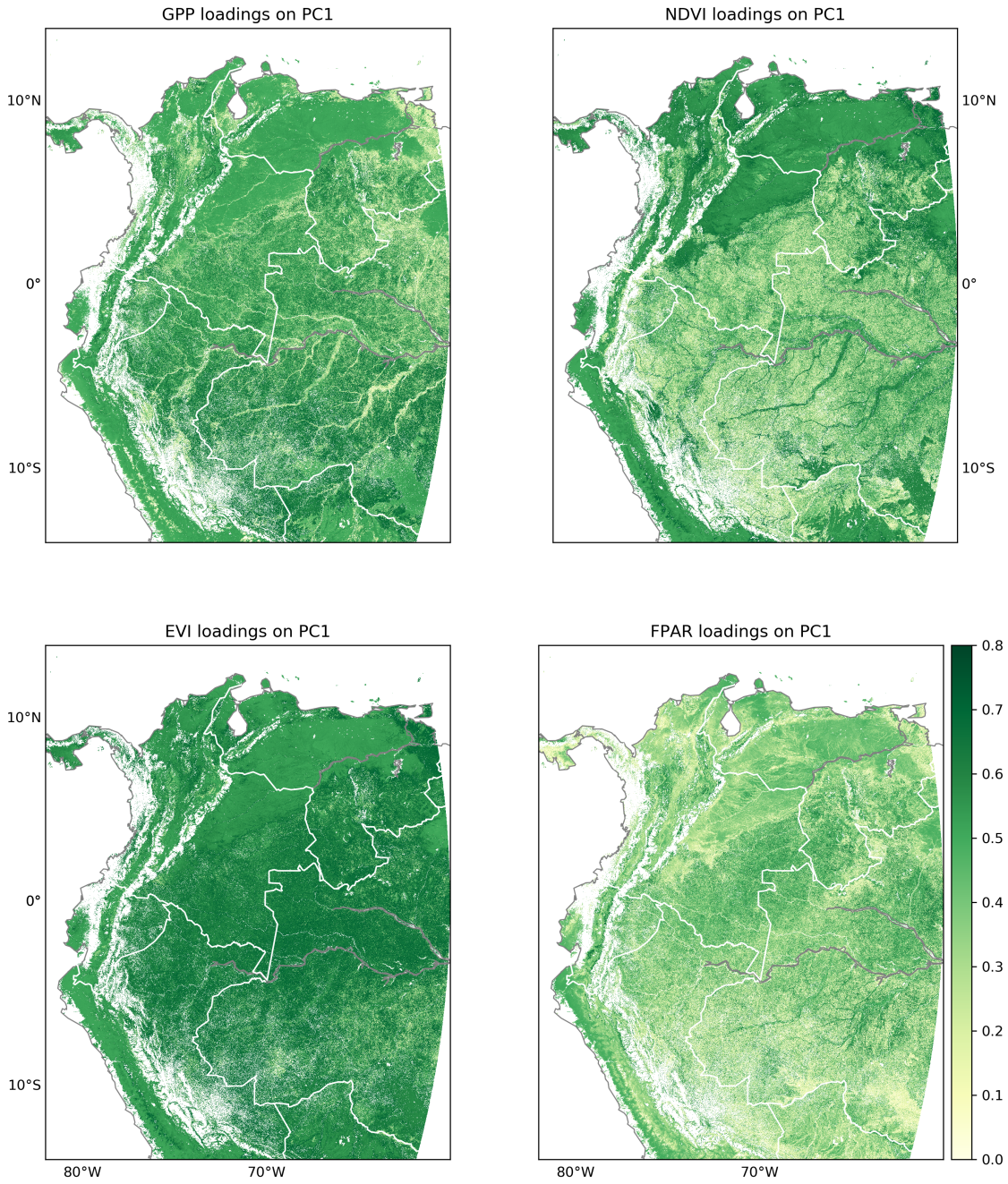



**Figure S3.** Loading values of vegetation variables for the first component of PCA. GPP: Gross Primary Productivity, NDVI: Normalized Difference Vegetation Index. EVI: Enhanced Vegetation Index. FPAR: Photosynthetically Active Radiation. National borders are delineated in withe. Color map shows absolute values.

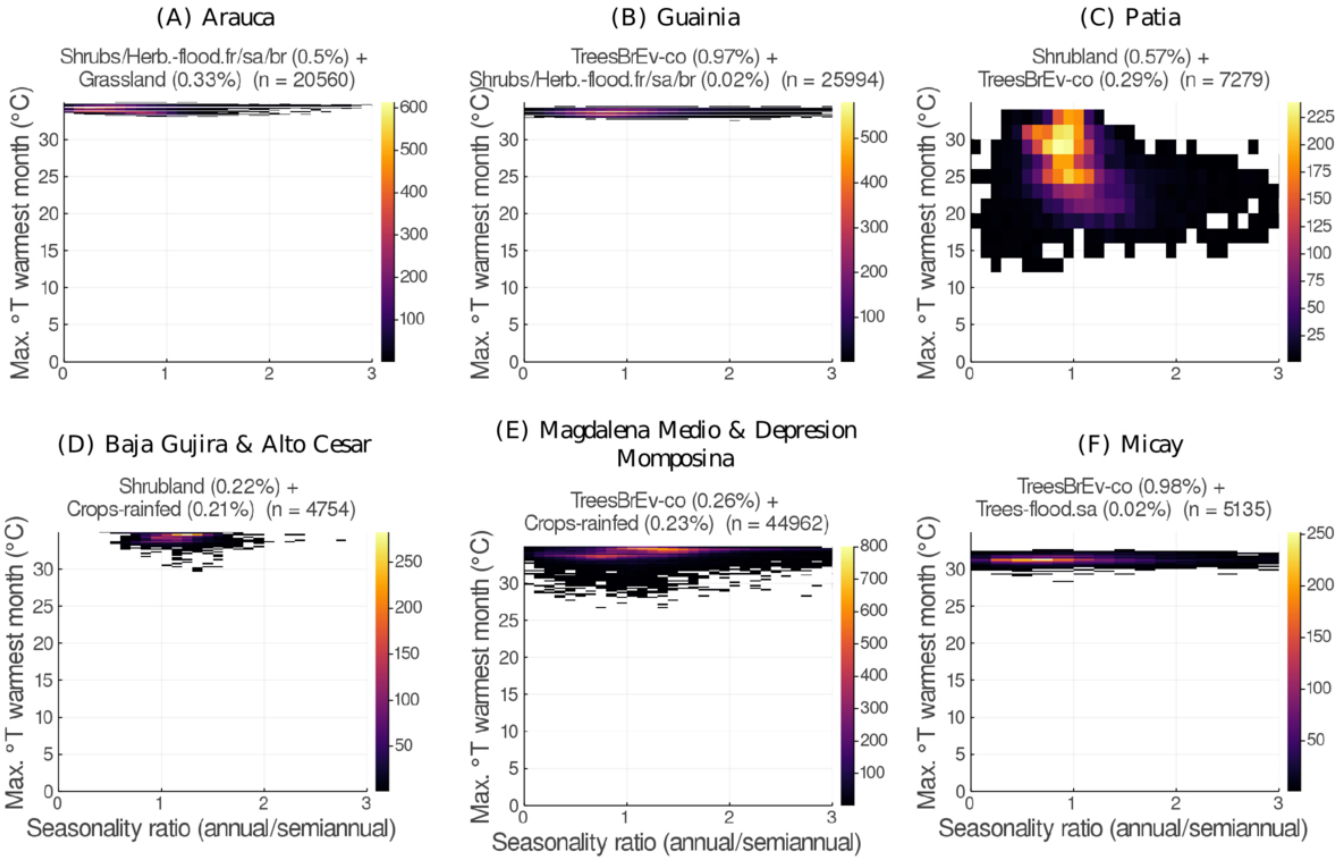



**Figure S4.** Seasonality ratio of annual and semiannual oscillation (x-axis) and maximum temperature of the warmest month (y-axis). Note that the color scheme range varies between figures. n is the total number of pixels in each biotic unit

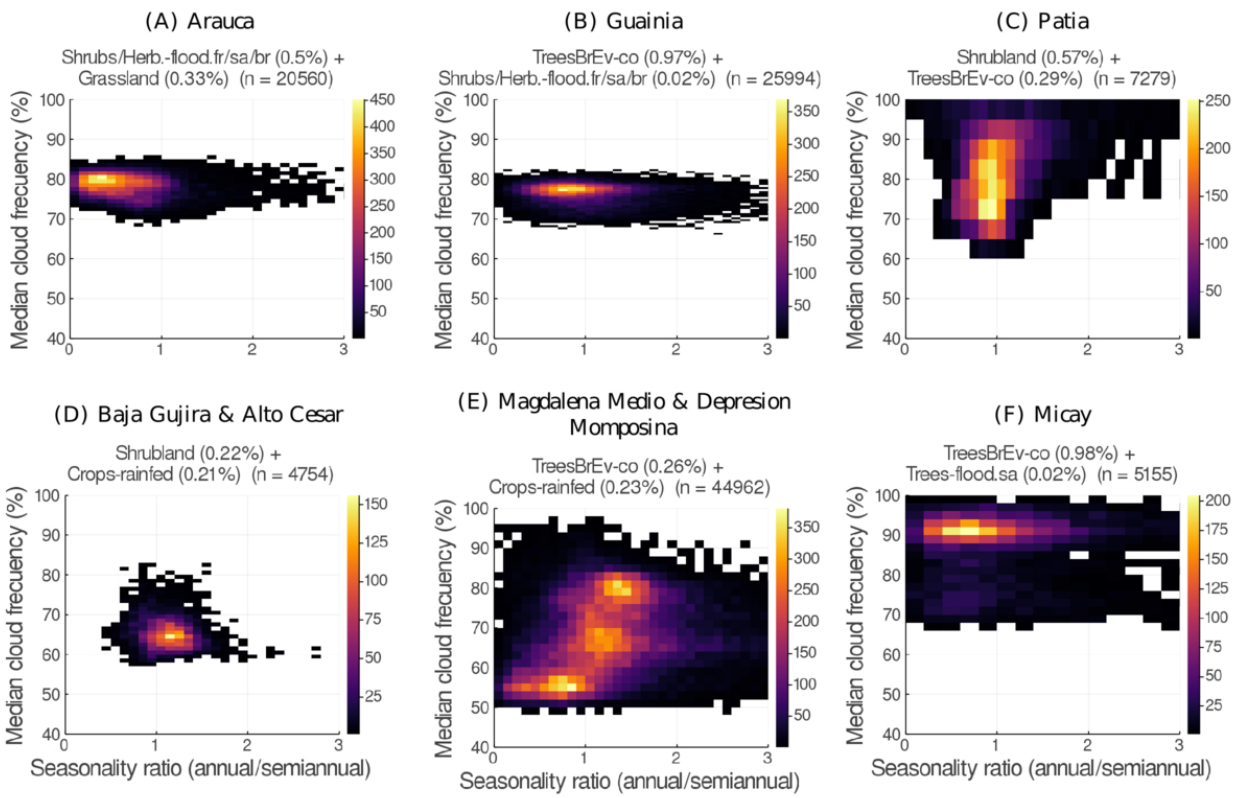



**Figure S5.** Seasonality ratio of annual and semiannual oscillation (x-axis) and annual median of clouds frequency (y-axis). Note that the color scheme range differs between plots. "n" is the total number of pixels in each biotic unit.

## 3 SCRIPTS

The following scripts were exported from Jupyter notebooks and correspond to the scripts specified in Table S7.

# Data access

# Script 01: Code to download the regional ESDL using Python

### Estupinan-Suarez, Gans, et al. (2021). A Regional Earth System Data Lab for Understanding Ecosystem Dynamics: An Example from Tropical South America. Front. Earth Sci. 9:613395. doi: 10.3389/feart.2021.613395

**Correspondence to: lestup@bgc-jena.mpg.de, linamaesu@gmail.com**

**GitHub repository: https://github.com/linamaes/Regional_ESDL (https://github.com/linamaes/Regional_ESDL)**

About the script:

- This code was written in jupyter notebooks
- This notebook exemplifies how to access the RegESDL. Here, we showcase how to download the RegESDL using s3fs in Python for local use

About the notebook:

- The notebook kernel is Python 3.7.1, and it uses the s3fs package version 0.5.2
- "#" comments in the code are intended to explain specific aspects of the coding
- New steps in workflows are introduced with bold headers
- For complete information of the datasets check the orignal data source documentation. Citations are available on the referred manuscript and supplementary tables.

June 2021, Max Planck Insitute for Biogeochemistry, Jena, Germany

# Import requiered packages

In [ ]:
```
import s3fs
```

# Commands used to download the RegESDL

In [ ]:
```
# Create a handle to the otc object store
s3 = s3fs.S3FileSystem(
    anon = True,
)
```

In [ ]:
```
# List of available colombia data cubes:
s3.ls('esdl-esdc-v2.0.1')
```

In [ ]:
```
# Commands used to download a cube
obs.get(obs.ls('esdl-esdc-v2.0.1/Cube_2019highColombiaCube_184x120x120.zarr'
),'./mylocalhighrescube', recursive=True);
```

Commands to access the data cube using xcube and then continue analysis using xarray:

In [ ]:
```
import xarray as xr
import xcube
```

In [ ]:
```
from xcube.core.dsio import open_dataset

ds = open_dataset("https://s3.eu-central-1.amazonaws.com/esdl-esdc-v2.0.1/Cube_2019highColombiaCube_184x120x120.zarr", s3_kwargs=dict(anon=True))
```

In [ ]:
```

```

# Computational Performance

# Script 03: Computational comparison of time series decomposition using different Data Cubes storage

### Estupinan-Suarez, Gans, et al. (2021). A Regional Earth System Data Lab for Understanding Ecosystem Dynamics: An Example from Tropical South America. Front. Earth Sci. 9:613395. doi: 10.3389/feart.2021.613395

**Correspondence to: lestup@bgc-jena.mpg.de, linamaesu@gmail.com**

**GitHub repository: https://github.com/linamaes/Regional_ESDL (https://github.com/linamaes/Regional_ESDL)**

This script illustrates the time difference on time series computing when using different cube storage versions. We provide two examples:

- Example 1: Time series are decomposed in different time-scales i.e. Longer-term, seasonal cycle and short(fast) oscillation, plus the trend
- Example 2: It finds the maximum value along a time period at pixel level

About the notebook

- It is written in Julia 1.3
- "#" comments in the code are intended to explain specific aspects of the coding
- New steps in the workflow are introduced with bold headers

March 2021, Max Planck Insitute for Biogeochemistry, Jena, Germany

# Load packages

In [1]:
```
using ESDL
```

In [2]:
```
using ESDLPlots
```

**Unable to load WebIO. Please make sure WebIO works for your Jupyter client. For troubleshooting, please see the WebIO/IJulia documentation (https://juliagizmos.github.io/WebIO.jl/latest/providers/ijulia/).**

# Set reading and writing paths

In [6]:
```
pathin = "../mypath/"# => Set your path
```

Out[6]:
```
"../mypath/"
```

# Load Cubes

In [4]:
```
# Cube storage version for temporal analysis
ctem = Cube(string(pathin,"/Cube_2020highColombiaCube_184x120x120.zarr/"))
```

Out[4]:
```
Collection of ZArray Cube with the following dimensions
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Time                 Axis with 782 Elements from 2001-01-05T00:00:00 to 2017-
12-31T00:00:00
Variable             Axis with 5 elements: lai fpar gross_primary_productivit
y evi ndvi
Total size: 168.85 GB
```

In [5]:
```
# Cube storage version for spatial analysis
cspa = Cube(string(pathin,"/Cube_2020highColombiaCube_1x3360x2760.zarr/"))
```

Out[5]:
```
Collection of ZArray Cube with the following dimensions
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Time                 Axis with 782 Elements from 2001-01-05T00:00:00 to 2017-
12-31T00:00:00
Variable             Axis with 5 elements: lai fpar gross_primary_productivit
y evi ndvi
Total size: 168.85 GB
```

## Settings for cube subset

In [7]:
```
yrstart = 2001
yrend = 2005
latsub =(0,1)
```

Out[7]:
```
(0, 1)
```

## Example 1: Time series decomposition

In [8]:
```
@time ctemfft = filterTSFFT(ctem[time=yrstart:yrend, lat=latsub])
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:02:55

186.832809 seconds (1.57 G allocations: 55.861 GiB, 6.66% gc time)
```

Out[8]:
```
Collection of ZArray Cube with the following dimensions
Time                 Axis with 230 Elements from 2001-01-05T00:00:00 to 2005-
12-31T00:00:00
Scale                Axis with 4 elements: Trend Long-Term Variability Annual
Cycle Fast Oscillations
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 120 Elements from 0.9961893499999996 to 0.0045
26649999999694
Variable             Axis with 5 elements: lai fpar gross_primary_productivit
y evi ndvi
Total size: 7.09 GB
```

In [9]:
```
@time cspafft = filterTSFFT(cspa[time=yrstart:yrend, lat=latsub])
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ESDL/skMpG/src/DAT/DAT.jl:608
┌ Warning: There are compressed caches misses, you may want to use a different cube chunking
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ESDL/skMpG/src/DAT/DAT.jl:609
Progress: 100%|████████████████████████████████████████| Time: 0:36:33

2194.006997 seconds (1.53 G allocations: 929.078 GiB, 4.11% gc time)
```

Out[9]:
```
Collection of ZArray Cube with the following dimensions
Time                 Axis with 230 Elements from 2001-01-05T00:00:00 to 2005-12-31T00:00:00
Scale                Axis with 4 elements: Trend Long-Term Variability Annual Cycle Fast Oscillations
Lon                  Axis with 2760 Elements from -82.99622135 to -60.00464665
Lat                  Axis with 120 Elements from 0.9961893499999996 to 0.004526649999999694
Variable             Axis with 5 elements: lai fpar gross_primary_productivity evi ndvi
Total size: 7.09 GB
```

## Example 2: Maximum values along time series

In [10]:
```
@time ctempax= mapslices(maximum, ctem[time=yrstart:yrend, lat=latsub], dims="time")
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:00:41

 43.233203 seconds (756.82 M allocations: 17.970 GiB, 9.68% gc time)
```

Out[10]:
```
In-Memory data cube with the following dimensions
Lon                  Axis with 2760 Elements from -82.99622135 to -60.00464665
Lat                  Axis with 120 Elements from 0.9961893499999996 to 0.004526649999999694
Variable             Axis with 5 elements: lai fpar gross_primary_productivity evi ndvi
Total size: 7.9 MB
```

In [11]:
```
@time cspamax = mapslices(maximum, cspa[time=yrstart:yrend, lat=latsub], dim
s="time")
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:608
┌ Warning: There are compressed caches misses, you may want to use a differe
nt cube chunking
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:609
Progress: 100%|████████████████████████████████████████| Time: 0:12:38

758.984782 seconds (754.09 M allocations: 249.939 GiB, 4.20% gc time)
```

Out[11]:
```
In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 120 Elements from 0.9961893499999996 to 0.0045
26649999999694
Variable            Axis with 5 elements: lai fpar gross_primary_productivit
y evi ndvi
Total size: 7.9 MB
```

In [ ]:

# Main figures

# Figure 4: Variance explained by principal components of vegetation variables

### Estupinan-Suarez, et al. (2021). A Regional Earth System Data Lab for Understanding Ecosystem Dynamics: An Example from Tropical South America. Front. Earth Sci. 9:613395. doi: 10.3389/feart.2021.613395

**Correspondence to: lestup@bgc-jena.mpg.de, linamaesu@gmail.com**

**GitHub repository: https://github.com/linamaes/Regional_ESDL (https://github.com/linamaes/Regional_ESDL)**

This script does the following:

- Gap fills and standarized data to zero mean and variance of 1
- Computes Principal Component Analysis (PCA) for vegetation variables i.e., GPP, NDVI, EVI, FAPAR
- Calculates the significance of each principal component (PC)
- Extracts loading values of each component
- Computes the mean seasonal cycle (MSC) and standard deviation seasonal cycle (SDSC) of the first PC

About the notebook:

- It is written in Julia 1.3
- "#" comments in the code are intended to explain specific aspects of the coding
- New steps in workflows are introduced with bold headers

April 2021, Max Planck Institute for Biogeochemistry, Jena, Germany

# Load packages

In [1]:
```
using ESDL
```

In [2]:
```
using ESDLPlots
```

**Unable to load WebIO. Please make sure WebIO works for your Jupyter client. For troubleshooting, please see the WebIO/IJulia documentation (https://juliagizmos.github.io/WebIO.jl/latest/providers/ijulia/).**

In [3]:
```
using CSV
```

In [4]:
```
using Missings
```

In [5]:
```
using Plots
```

In [6]:
```
using MultivariateStats
```

In [7]:
```
using OnlineStats
```

In [8]:
```
using DelimitedFiles
```

In [9]:
```
using Distributed
```

In [10]:
```
using NPZ
```

In [11]:
```
using Statistics
```

In [12]:
```
gr(size=(600,400))
default(fmt = :png)
```

## Load RegESDL for temporal analysis

In [13]:
```
pathin = "/my_path/.../"
```

Out[13]:
```
"/my_path/.../"
```

In [15]:
```
# High spatial resolution
c = Cube(string(pathin, "Cube_2020highColombiaCube_184x120x120.zarr/"))

# Low spatial resolution
# c = Cube(string(pathin, "Cube_2020lowColombiaCube_184x60x60.zarr/"))
```

Out[15]:
```
Collection of ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Time                Axis with 782 Elements from 2001-01-05T00:00:00 to 2017-
12-31T00:00:00
Variable            Axis with 5 elements: lai fpar gross_primary_productivit
y evi ndvi
Total size: 168.85 GB
```

## Parameters for data subsets

In [16]:
```
csub = c[time=2001:2014, var=["gross_primary_productivity", "ndvi",
        "evi","fpar"]]#
```

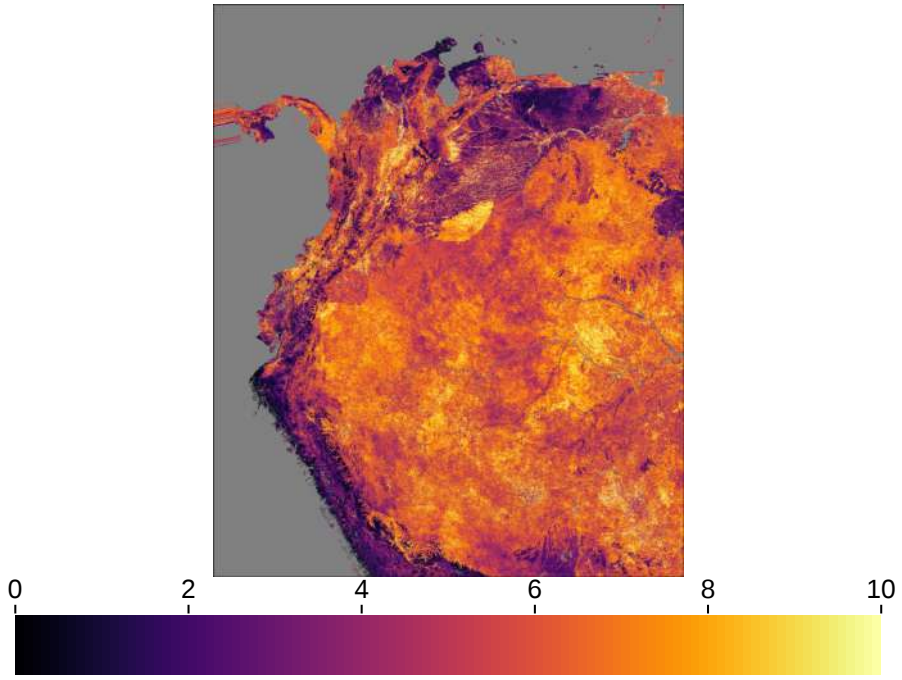
Out[16]:
```
Collection of ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Time                Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Variable            Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

In [17]:
```
# Set time for plotting
t = Date(2014, 1, 31)
```

Out[17]:
```
2014-01-31
```

In [18]:
```
plotMAP(csub, time=t, variable="gross_primary_productivity", dmin=0, dmax=10
)
```

Out[18]:




## Load mask

In [19]:
```
pathcsv = "/my_mask_path/.../"
```

Out[19]:
```
"/my_mask_path/.../"
```

In [20]:
```
pathcsv = "/Net/Groups/BGI/people/lestup/ESDLreg/output/csv/"
```

Out[20]:
```
"/Net/Groups/BGI/people/lestup/ESDLreg/output/csv/"
```

In [21]:
```
maskin = CSV.read(string(pathcsv, "msc_missings_mask.csv"), header=false);
```

In [22]:
```
maskin = Matrix(maskin);
```

In [23]:
```
mask = map(x-> isnan(x) ? missing : x==0 ? missing : x, maskin' |> Array);
```

In [24]:
```
csub
```

Out[24]:
```
Collection of ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Time                Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Variable            Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

In [25]:
```
# lonaxis = RangeAxis("Lon", lon)
# lataxis = RangeAxis("Lat", lat)
lonaxis = csub.cubeaxes[1]
lataxis = csub.cubeaxes[2]
```

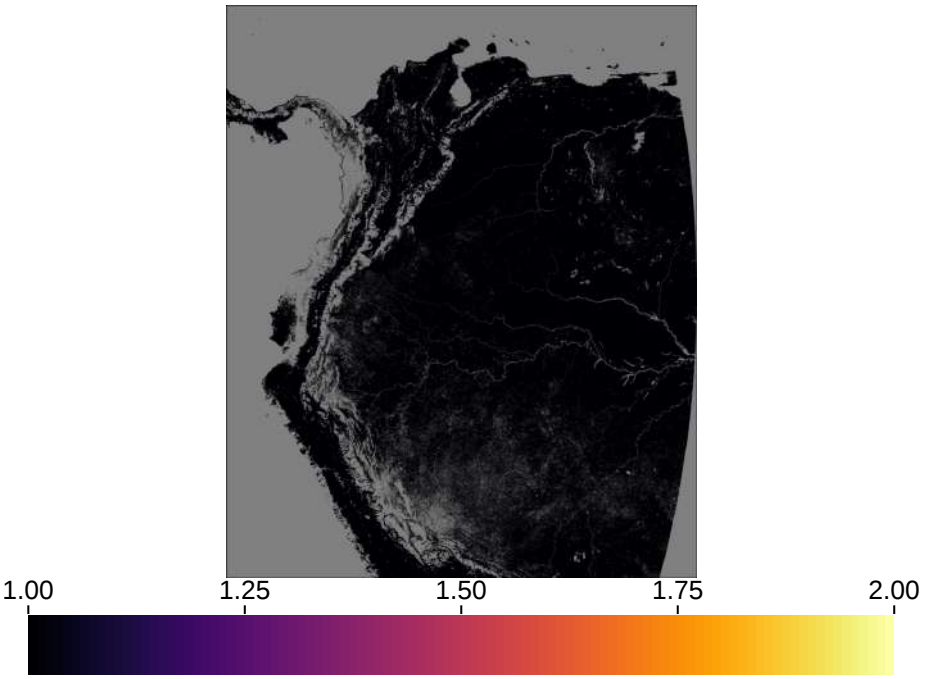
Out[25]:
```
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
```

In [26]:
```
# Create a cube from the mask
cmask = CubeMem(CubeAxis[lonaxis, lataxis], mask)
```

Out[26]:
```
In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 79.6 MB
```

In [27]:
```
plotMAP(cmask)
```

Out[27]:




## Settings for Parallel Processing

In [28]:
```
workers()
```

Out[28]:
```
1-element Array{Int64,1}:
 1
```

In [29]:
```
## add workers
addprocs(6)
```

Out[29]:
```
6-element Array{Int64,1}:
 2
 3
 4
 5
 6
 7
```

## Define missings

In [30]:
```
csub
```

Out[30]:
```
Collection of ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Time                Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Variable            Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

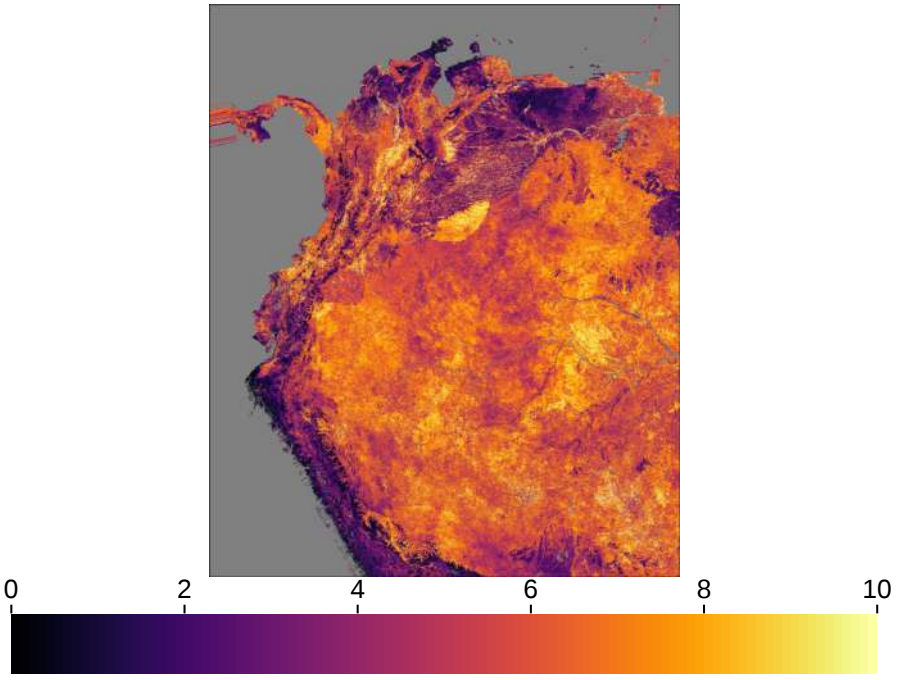
In [31]:
```
csub2 = map(x-> ismissing(x) ? missing : x==255.0 ? missing : x==65335.0 ? m
issing : x, csub)
```

Out[31]:
```
Transformed cube Collection of ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Time                Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Variable            Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

In [32]:
```
#csub2 = mapCube(maskout, csub[lat=lat, lon=lon], mask, indims=indims, outdi
ms=outdims)
```

In [33]:
```
plotMAP(csub2, time=t, variable="gross_primary_productivity", dmin=0, dmax=1
0)
```

Out[33]:




## Gap-filling

In [34]:
```
#cfill = gapFillMSC(csub)
@time cfill = gapFillMSC(csub2)
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:05:25

365.476447 seconds (26.38 M allocations: 1.306 GiB, 0.08% gc time)
```

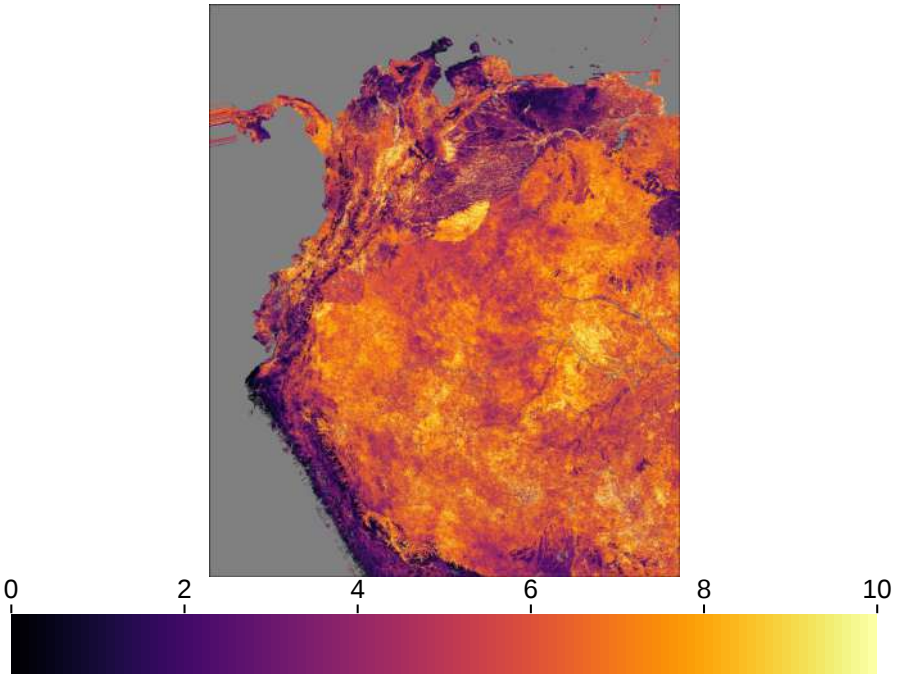
Out[34]:
```
Collection of ZArray Cube with the following dimensions
Time                 Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Variable             Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

In [35]:
```
cfill
```

Out[35]:
```
Collection of ZArray Cube with the following dimensions
Time                 Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Variable             Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

In [36]:
```
plotMAP(cfill, time=t, variable="gross_primary_productivity", dmin=0, dmax=1
0)
```

Out[36]:




## Standarized function

In [37]:
```
@everywhere using OnlineStats
```

In [38]:
```
# Variables are standardized to zero mean and variance of 1

@everywhere function stdFx(xout, xin)
    xout[:] .= map(x -> (x - OnlineStats.mean(xin))/sqrt(var(xin)), xin)
end
```

In [39]:
```
indimsstd = InDims("Time")
outdimsstd = OutDims("Time")
```

Out[39]:
```
OutDims((ESDL.Cubes.Axes.ByName("Time"),), (), zero, identity, :auto, false,
AsArray(), :input, Zarr.NoCompressor(), "", false, 1)
```

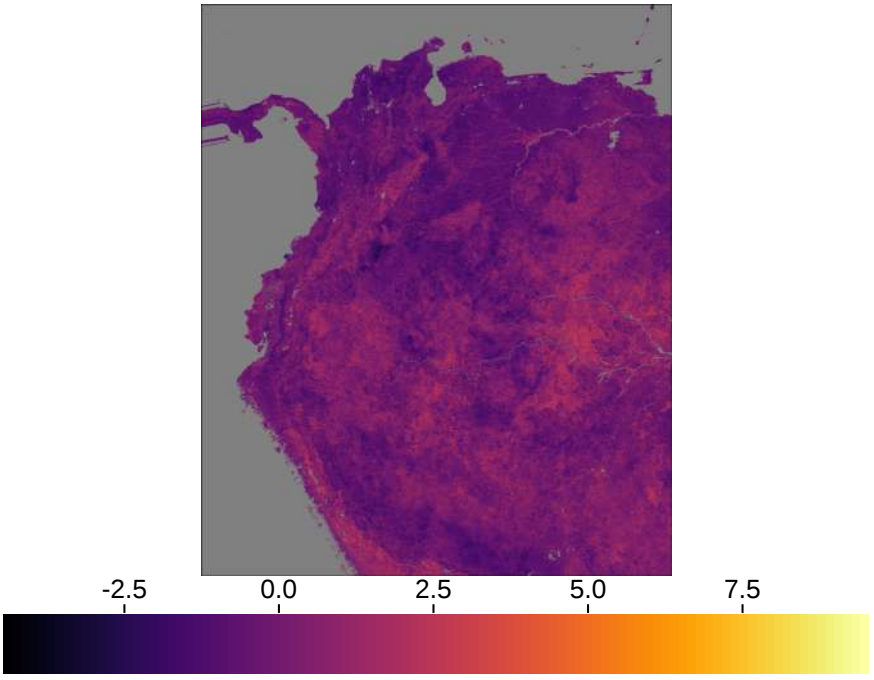
In [40]:
```
#@time
cstd = mapCube(stdFx, cfill, indims = indimsstd, outdims = outdimsstd)
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:25:48
```

Out[40]:
```
Collection of ZArray Cube with the following dimensions
Time                 Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Variable             Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

In [41]:
```
plotMAP(cstd, time=t, variable="gross_primary_productivity")
```

Out[41]:




## PC significance and loading values from PCA

In [42]:
```
# addprocs(8)
```

In [43]:
```
workers();
```

Out[43]:
```
6-element Array{Int64,1}:
 2
 3
 4
 5
 6
 7
```

In [44]:
```
# rmprocs(workers())
```

In [45]:
```
@everywhere using MultivariateStats
```

In [46]:
```
@everywhere using ESDL
```

In [47]:
```
@everywhere function pcaSigFx(xout, xin0::Any, loopvars, maxoutdim=4, method
=:svd)
    #@show loopvars
    if any(ismissing.(xin0))
        return missing
    else
       #@show size(xin0)
       xin = convert(Array{Float32}, xin0' |> Matrix)
        # projection(pca) give the rotation matrix
        xin_pca = fit(PCA, xin; maxoutdim=maxoutdim,  method=method, pratio=
1.0)
       #@show size(xout)
        xout[:,1] .= (principalvars(xin_pca) ./ tprincipalvar(xin_pca))
        xout[:,2:4] .= projection(xin_pca)[:,1:3]
        return xout

    end
end
```

In [48]:
```
sigaxis = CategoricalAxis("info_PCA", ["PC_Significance","LoadingPC1", "Load
ingPC2",
        "LoadingPC3"])#,, "LoadingPC4"

varsigaxis = CategoricalAxis("Components_or_Var", ["sigPC_var1","sigPC_var2"
,"sigPC_var3", "sigPC_var4"]) #,"sigPC_var4"
```

Out[48]:
```
Components_or_Var   Axis with 4 elements: sigPC_var1 sigPC_var2 sigPC_var3 s
igPC_var4
```

In [49]:
```
indims = InDims("Time","Variable")
outdims = OutDims(varsigaxis, sigaxis)
```

Out[49]:
```
OutDims((ESDL.Cubes.Axes.ByValue(Components_or_Var   Axis with 4 elements: s
igPC_var1 sigPC_var2 sigPC_var3 sigPC_var4 ), ESDL.Cubes.Axes.ByValue(info_P
CA            Axis with 4 elements: PC_Significance LoadingPC1 LoadingPC2 Lo
adingPC3 )), (), zero, identity, :auto, false, AsArray(), :input, Zarr.NoCom
pressor(), "", false, 1)
```

In [50]:
```
cstd
```

Out[50]:
```
Collection of ZArray Cube with the following dimensions
Time                 Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Variable             Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

In [51]:
```
########################################
# NOTE for axes interpretation:
# 'Components_or_Var' has a different meaning based on the 'info_PCA' class,
as follows:
#  For 'PC_Significance' the values are  significance of PC1, PC2, PC3 and P
C4 respectively
#  For 'LoadingsPC1/2/3' the values are the loadings of variable 1, 2, 3, an
d 4 respectively
#  Check the variables order from the input data.
########################################

@time pcasiglod = mapCube(pcaSigFx, cstd, 4, indims=indims, outdims=outdims)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:01:37

100.683267 seconds (4.34 M allocations: 214.748 MiB, 0.07% gc time)
```

Out[51]:
```
ZArray Cube with the following dimensions
Components_or_Var    Axis with 4 elements: sigPC_var1 sigPC_var2 sigPC_var3 s
igPC_var4
info_PCA             Axis with 4 elements: PC_Significance LoadingPC1 Loading
PC2 LoadingPC3
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 707.52 MB
```

In [52]:
```
# saveCube(pcasiglod, "pcaComSigLoadings1km2014_cube2020")
```

In [53]:
```
indims = InDims("Lon","Lat")
outdims = OutDims("Lon","Lat")
```

Out[53]:
```
OutDims((ESDL.Cubes.Axes.ByName("Lon"), ESDL.Cubes.Axes.ByName("Lat")), (),
zero, identity, :auto, false, AsArray(), :input, Zarr.NoCompressor(), "", fa
lse, 1)
```

In [54]:
```
rmprocs(workers())
```

Out[54]:
```
Task (done) @0x00007f97a5a19870
```

## Mask data

In [55]:
```
indims = InDims("Lon","Lat")
outdims = OutDims("Lon","Lat")
```

Out[55]:
```
OutDims((ESDL.Cubes.Axes.ByName("Lon"), ESDL.Cubes.Axes.ByName("Lat")), (),
zero, identity, :auto, false, AsArray(), :input, Zarr.NoCompressor(), "", fa
lse, 1)
```

In [56]:
```
# This function assigns values of missing to low quality pixels

function maskout(xout, xin, mask)
    xout[:] = xin.*mask
end
```

Out[56]:
```
maskout (generic function with 1 method)
```

In [57]:
```
pcasiglodout = mapCube(maskout, pcasiglod, mask, indims=indims, outdims=outdims)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ESDL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:00:12
```

Out[57]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.00464665
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Components_or_Var   Axis with 4 elements: sigPC_var1 sigPC_var2 sigPC_var3 sigPC_var4
info_PCA            Axis with 4 elements: PC_Significance LoadingPC1 LoadingPC2 LoadingPC3
Total size: 707.52 MB
```

In [58]:
```
# saveCube(pcasiglodout, "pcaComSigLoadings1km2014_mask")
```

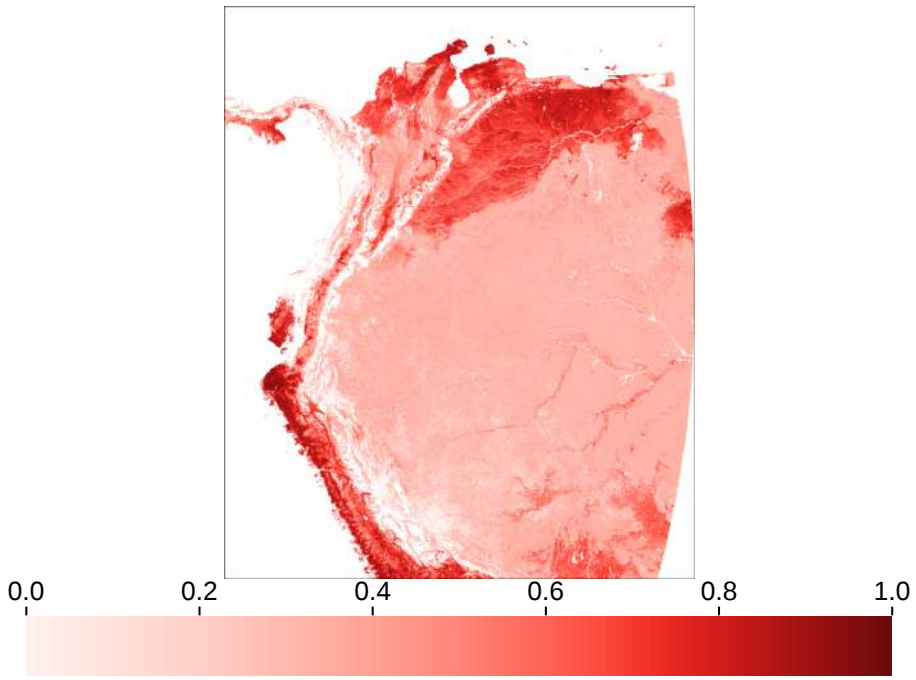
## Plotting PCs explained variance - Figure 3 A, B, C.

In [99]:
```
pcasigm = pcasiglodout[info_PCA="PC_Significance"]
```

Out[99]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.00464665
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Components_or_Var   Axis with 4 elements: sigPC_var1 sigPC_var2 sigPC_var3 sigPC_var4
Total size: 176.88 MB
```

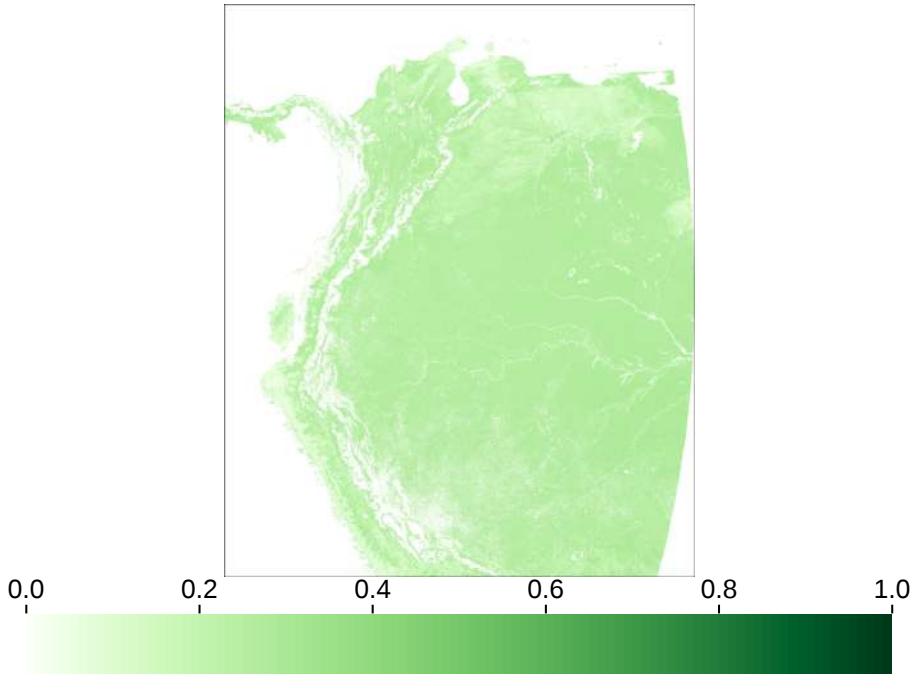
In [100]:
```
plotMAP(pcasigm[Components="sigPC_var1"], colorm=colormap("reds"), dmin=0, dmax=1,
    oceancol=colorant"white",
    misscol=colorant"white",
    dpi=150)
```

Out[100]:



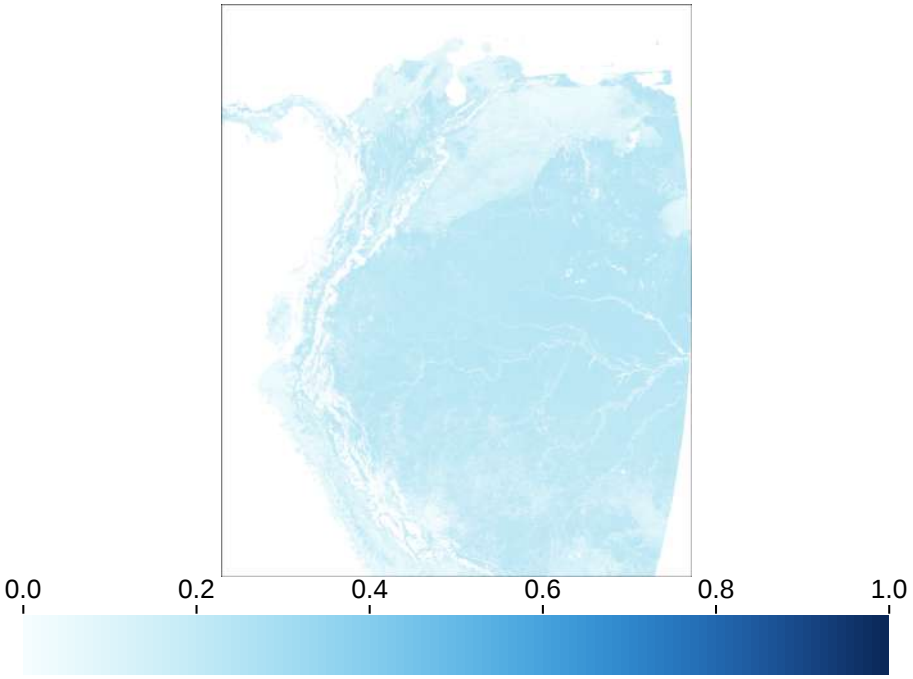

In [101]:
```
plotMAP(pcasigm[Components="sigPC_var2"], colorm=colormap("greens"), dmin=0, dmax=1,
    oceancol=colorant"white",
    misscol=colorant"white")
```

Out[101]:



In [102]:
```
plotMAP(pcasigm[Components="sigPC_var3"], colorm=colormap("blues"), dmin=0,
dmax=1,
    oceancol=colorant"white",
    miscol=colorant"white") # misscol=colorant"white",
```

Out[102]:




In [103]:
```
minimum(skipmissing(pcasigm[:,:,1])), maximum(skipmissing(pcasigm[:,:,1]))
```

Out[103]:
```
(0.25313136f0, 0.96207017f0)
```

In [104]:
```
minimum(skipmissing(pcasigm[:,:,2])), maximum(skipmissing(pcasigm[:,:,2]))
```

Out[104]:
```
(0.019068716f0, 0.46054468f0)
```

In [105]:
```
minimum(skipmissing(pcasigm[:,:,3])), maximum(skipmissing(pcasigm[:,:,3]))
```

Out[105]:
```
(0.010792308f0, 0.27545634f0)
```

In [106]:
```
quantile(skipmissing(pcasigm[:,:,1]),[0.01, .99])
```

Out[106]:
```
2-element Array{Float64,1}:
 0.2804080545902252
 0.8633400797843933
```

In [107]:
```
quantile(skipmissing(pcasigm[:,:,2]),[0.01, .99])
```

Out[107]:
```
2-element Array{Float64,1}:
 0.07589496798813343
 0.3134177041053774
```

In [108]:
```
quantile(skipmissing(pcasigm[:,:,3]),[0.01, .91])
```

Out[108]:
```
2-element Array{Float64,1}:
 0.03779849950224161
 0.24152633115649225
```

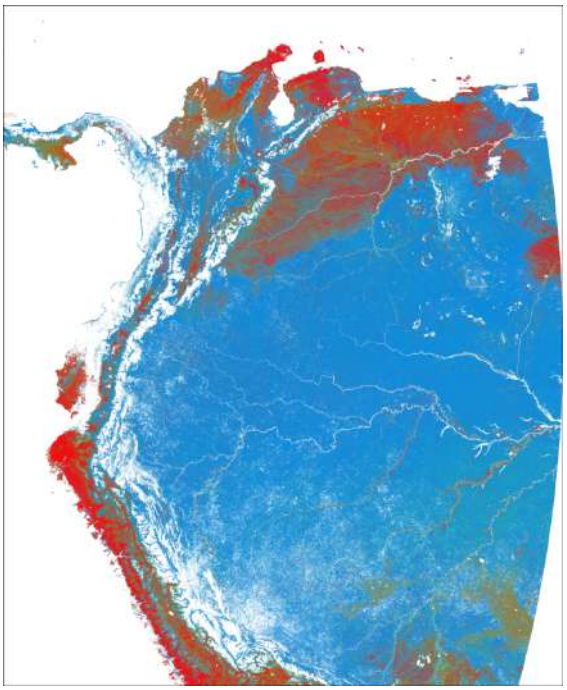
## Plotting Figure 4D - RGB map

In [68]:
```
pcasigm = pcasiglodout[info_PCA="PC_significance"]
```

Out[68]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Components_or_Var   Axis with 4 elements: sigPC_var1 sigPC_var2 sigPC_var3 s
igPC_var4
Total size: 176.88 MB
```

In [69]:
```
p1 = plotMAPRGB(pcasigm, rgbaxis="Components",
    oceancol=colorant"white",
    misscol=colorant"white",
    c1="sigPC_var1",
    c2="sigPC_var2",
    c3="sigPC_var3",
    cType=RGB)
```

Out[69]:



## Data projected in the PCA space

In [70]:
```
addprocs(6);
```

Out[70]:
```
6-element Array{Int64,1}:
  8
  9
 10
 11
 12
 13
```

In [71]:
```
workers();
```

Out[71]:
```
6-element Array{Int64,1}:
  8
  9
 10
 11
 12
 13
```

In [72]:
```
@everywhere using MultivariateStats
```

In [73]:
```
@everywhere function pcaFx(xout, xin0::Any, maxoutdim=maxd, method=:svd)
    if any(ismissing.(xin0))
        return missing
    else
       xin = convert(Array{Float32}, xin0' |> Matrix)
        # projection(pca) give the rotation matrix
        xin_pca = fit(PCA, xin; maxoutdim=maxoutdim,  method=method, pratio=
1.0)
        t = transform(xin_pca, xin)
        xout .= missing
        for i in 1:min(size(xout,2),size(t,1))
            xout[:,i] .= t[i,:]
        end
    end
end
```

In [74]:
```
axispca = CategoricalAxis("Components", ["C1","C2","C3","C4"]) #,"C5"
```

Out[74]:
```
Components          Axis with 4 elements: C1 C2 C3 C4
```

In [75]:
```
indimspca = InDims("Time","Variable")
outdimspca = OutDims("Time",axispca)
```

Out[75]:
```
OutDims((ESDL.Cubes.Axes.ByName("Time"), ESDL.Cubes.Axes.ByValue(Components
Axis with 4 elements: C1 C2 C3 C4 )), (), zero, identity, :auto, false, AsAr
ray(), :input, Zarr.NoCompressor(), "", false, 1)
```

In [76]:
```
cstd
```

Out[76]:
```
Collection of ZArray Cube with the following dimensions
Time                Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Variable            Axis with 4 elements: gross_primary_productivity ndvi ev
i fpar
Total size: 111.24 GB
```

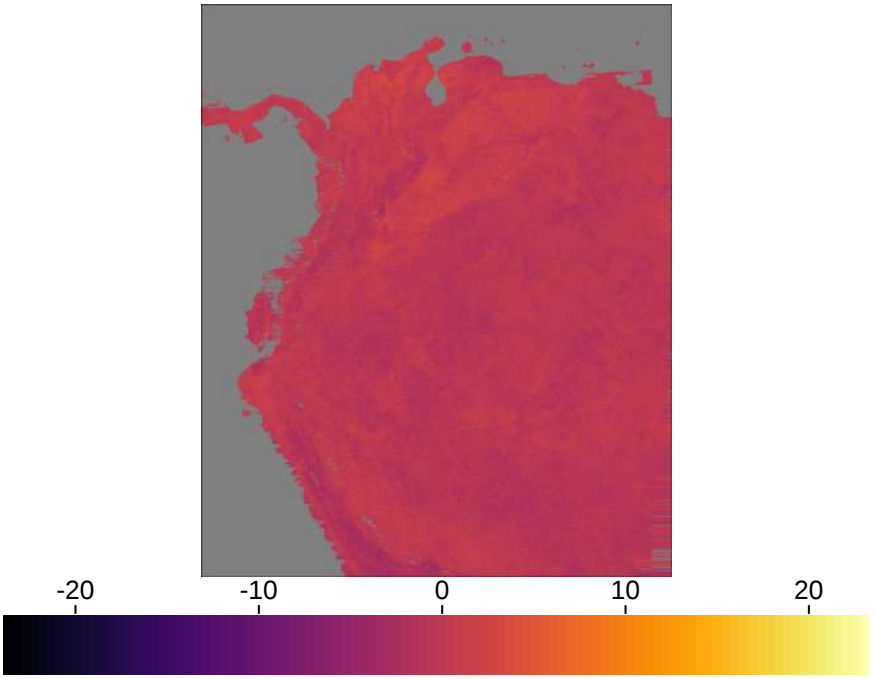
In [77]:
```
#pcacom = mapCube(pcaFx, cstd, 5, indims=indims, outdims=outdims)
@time pcacom = mapCube(pcaFx, cstd, 4, indims=indimspca, outdims=outdimspca)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:03:42

250.192142 seconds (711.13 k allocations: 33.416 MiB)
```

Out[77]:
```
ZArray Cube with the following dimensions
Time                Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Components          Axis with 4 elements: C1 C2 C3 C4
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 111.24 GB
```

In [78]:
```
plotMAP(pcacom, time=t, Component="C1")
```

Out[78]:




In [79]:
```
# saveCube(pcacom[Components="C1"],"pcacomstdTS1pc1km2014_mask")
```

## Get MSC

In [80]:
```
@time cmsc = getMSC(pcacom[Components="C1"])
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:00:20

 24.186404 seconds (3.50 M allocations: 175.855 MiB)
```

Out[80]:
```
ZArray Cube with the following dimensions
MSC                  Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 1.99 GB
```

In [81]:
```
@everywhere function maskout(xout, xin, mask)
    xout[:] = xin.*mask
end
```

In [82]:
```
indims = InDims("Lon","Lat")
outdims = OutDims("Lon","Lat")
```

Out[82]:
```
OutDims((ESDL.Cubes.Axes.ByName("Lon"), ESDL.Cubes.Axes.ByName("Lat")), (),
zero, identity, :auto, false, AsArray(), :input, Zarr.NoCompressor(), "", fa
lse, 1)
```

In [83]:
```
workers();
```

Out[83]:
```
6-element Array{Int64,1}:
  8
  9
 10
 11
 12
 13
```

In [84]:
```
rmprocs(workers())
```

Out[84]:
```
Task (done) @0x00007f96d0ae2bf0
```

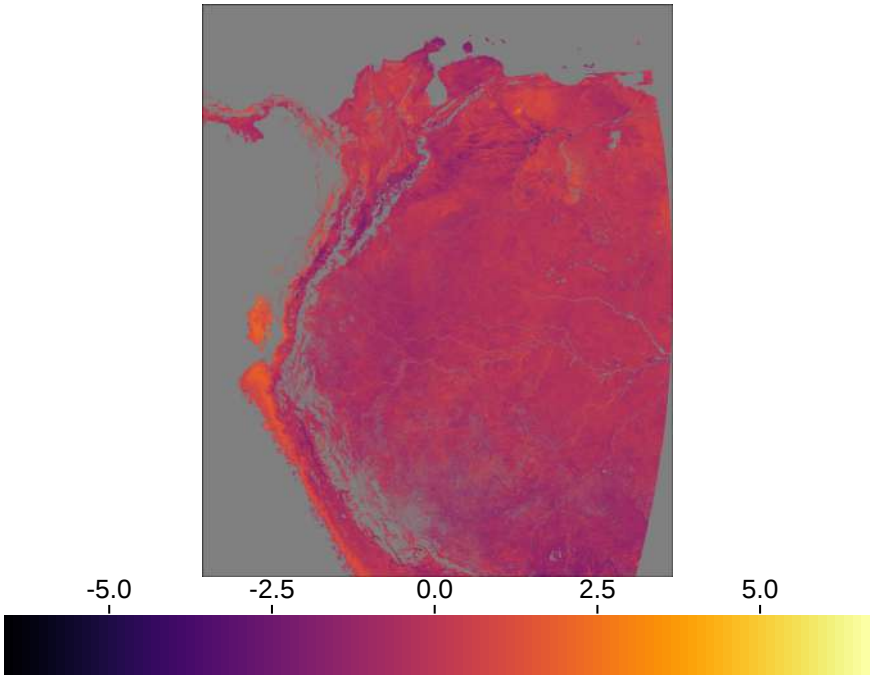
In [85]:
```
@time cmscm = mapCube(maskout, cmsc, mask, indims=indims, outdims=outdims)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ESDL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:00:39

 41.188557 seconds (8.18 M allocations: 37.019 GiB, 4.89% gc time)
```

Out[85]:
```
ZArray Cube with the following dimensions
Lon                  Axis with 2760 Elements from -82.99622135 to -60.00464665
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
MSC                  Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-12-27T00:00:00
Total size: 1.99 GB
```

In [86]:
```
plotMAP(cmscm, MSC=Date(1900, 12, 27))
```

Out[86]:




In [87]:
```
# saveCube(cmscm, "pcacom1msc1km2014_mask")
```

## Calculate standard deviation of Seasonal Cycle

In [88]:
```
axmsc = cmsc.axes[1]
```

Out[88]:
```
MSC                  Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-12-27T00:00:00
```

In [89]:
```
@everywhere function getSDSC(xout, xin, timeax)
    any(ismissing, xin) && return missing
    timedoy = map(x->Dates.dayofyear(x), timeax)
    #xout = zeros(46)
    for i in 1:46
        idx = findall(x-> x==timedoy[i], timedoy)
        xout[i] = std(map(x->xin[x], idx))
    end
    return xout
end
```

In [90]:
```
indimssdsc = InDims("Time")
outdimssdsc = OutDims(axmsc)
```

Out[90]:
```
OutDims((ESDL.Cubes.Axes.ByValue(MSC                  Axis with 46 Elements f
rom 1900-01-01T00:00:00 to 1900-12-27T00:00:00),), (), zero, identity, :aut
o, false, AsArray(), :input, Zarr.NoCompressor(), "", false, 1)
```

In [91]:
```
@everywhere using Dates, OnlineStats
```

In [92]:
```
tall = collect(pcacom.axes[1].values);
```

In [93]:
```
pcacom[Components="C1"]
```

Out[93]:
```
ZArray Cube with the following dimensions
Time                 Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 27.81 GB
```

In [94]:
```
@time csdsc = mapCube(getSDSC, pcacom[Components="C1"], tall, indims=indimss
dsc, outdims=outdimssdsc)
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:04:45

286.607642 seconds (4.28 G allocations: 359.293 GiB, 20.42% gc time)
```

Out[94]:
```
ZArray Cube with the following dimensions
MSC                  Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 1.99 GB
```

In [95]:
```
rmprocs(workers())
```

```
┌ Warning: rmprocs: process 1 not removed
└ @ Distributed /buildworker/worker/package_linux64/build/usr/share/julia/st
dlib/v1.3/Distributed/src/cluster.jl:1015
```

Out[95]:
```
Task (done) @0x00007f96cdac35b0
```

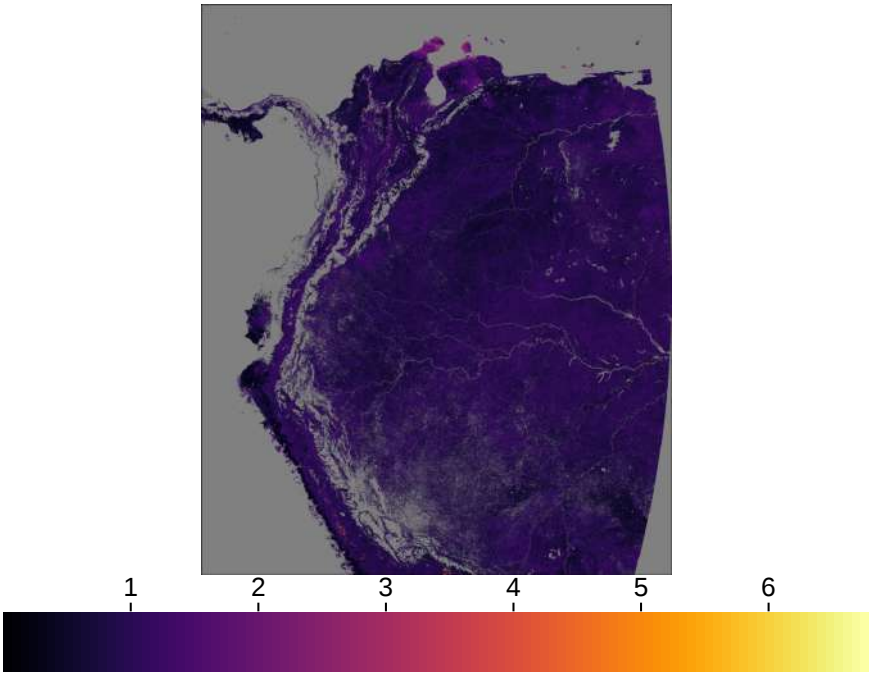
In [96]:
```
@time csdscm = mapCube(maskout, csdsc, mask, indims=indims, outdims=outdims)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:00:35

 35.746138 seconds (3.19 M allocations: 36.782 GiB, 2.32% gc time)
```

Out[96]:
```
ZArray Cube with the following dimensions
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
MSC                  Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Total size: 1.99 GB
```

In [97]:
```
plotMAP(csdscm, MSC=Date(1900, 12, 27))
```

Out[97]:




In [98]:
```
# saveCube(csdscm,"pcacom1sdsc1km2014_mask")
```

# Main figures

# Figure 5: Histograms of variance explained by the first three principal components (PCs)¶

## Estupinan-Suarez, et al. (2021). A Regional Earth System Data Lab for Understanding Ecosystem Dynamics: An Example from Tropical South America. Front. Earth Sci. 9:613395. doi: 10.3389/feart.2021.613395

**Correspondence to: lestup@bgc-jena.mpg.de, linamaesu@gmail.com**

**GitHub repository: https://github.com/linamaes/Regional_ESDL (https://github.com/linamaes/Regional_ESDL)**

This script does the following:

- Plots histograms of PCs explained variance by land cover types

About the notebook:

- It is written in Julia 1.3
- "#" comments in the code are intended to explain specific aspects of the coding
- New steps in workflows are introduced with bold headers

March 2021, Max Planck Institute for Biogeochemistry, Jena, Germany

## Activate environment, loading packages ¶

In [1]:
```
using OnlineStats
```

In [2]:
```
using ESDL
```

In [3]:
```
using ESDLPlots
```

**Unable to load WebIO. Please make sure WebIO works for your Jupyter client. For troubleshooting, please see the WebIO/IJulia documentation (https://juliagizmos.github.io/WebIO.jl/latest/providers/ijulia/).**

In [4]:
```
using CSV
```

In [5]:
```
using Dates
```

In [6]:
```
using NPZ
```

In [7]:
```
using NetCDF
```

In [8]:
```
using OnlineStats
```

In [9]:
```
using Plots
```

In [10]:
```
using DelimitedFiles
```

In [11]:
```
gr(size=(600,400))
default(fmt = :png)
```

# Load data

## PCA components significance 1km

In [12]:
```
pcasigin = loadCube("pcaComSigLoadings1km2014_qamask")
```

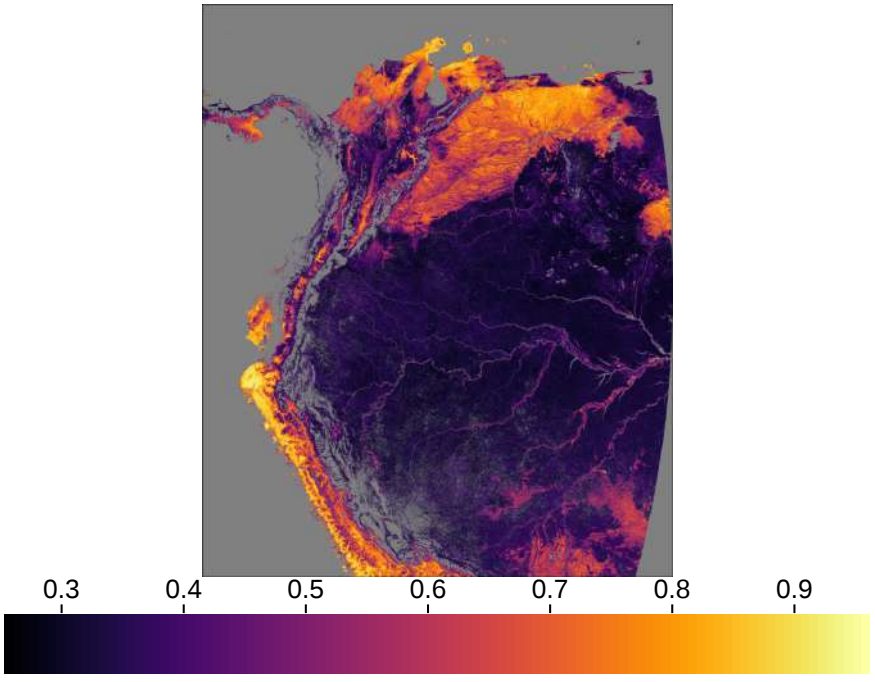
Out[12]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Components_or_Var   Axis with 4 elements: sigPC_var1 sigPC_var2 sigPC_var3 s
igPC_var4
info_PCA            Axis with 4 elements: PC_Significance LoadingPC1 Loading
PC2 LoadingPC3
Total size: 707.52 MB
```

In [13]:
```
pcasig = pcasigin[info_PCA="PC_Significance"]
```

Out[13]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Components_or_Var   Axis with 4 elements: sigPC_var1 sigPC_var2 sigPC_var3 s
igPC_var4
Total size: 176.88 MB
```

In [48]:
```
plotMAP(pcasig[Components_or_Var ="sigPC_var1"])
```

Out[48]:




## Load land cover map from ESA from a NetCDF file

In [15]:
```
pathin1 = "/my_path_in/folder1/"

pathin2 = "/my_path_in/folder2/"
```

Out[15]:
```
"/my_path_in/folder2/"
```

In [17]:
```
clcin = ncread(string(pathin1, "/LC_ESA/lc_esa2014.nc"),"Land.cover.class.defined.in.LCCS");
```

In [18]:
```
clcmis = map(x->x==210.0 ? missing : x, clcin);
```

In [19]:
```
pcasig.axes[1]
```

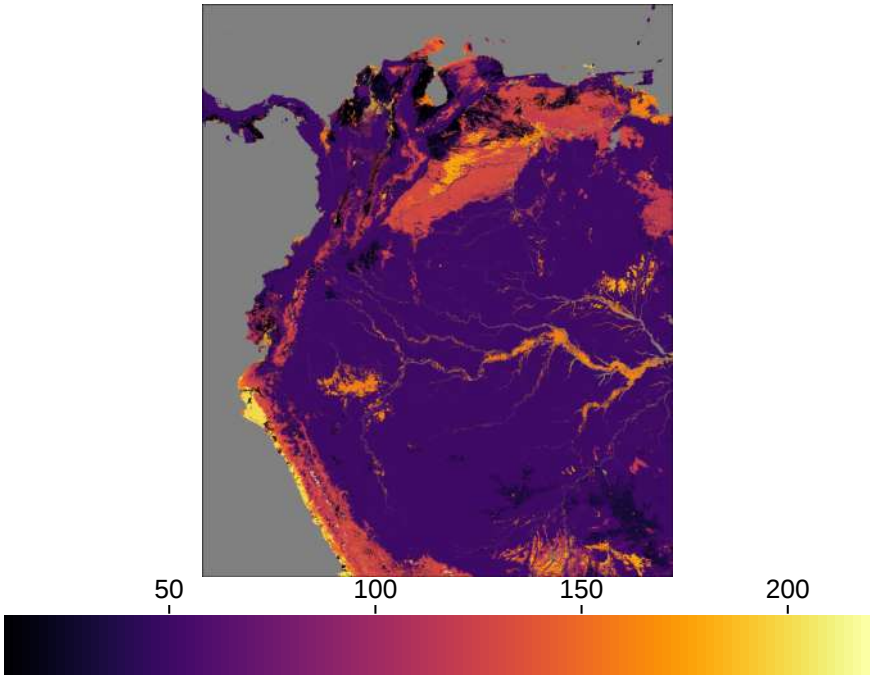
Out[19]:
```
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
```

In [20]:
```
clc = CubeMem(CubeAxis[pcasig.axes[1], pcasig.axes[2]], clcmis)
```

Out[20]:
```
In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 44.22 MB
```

In [21]:
```
plotMAP(clc)
```

Out[21]:

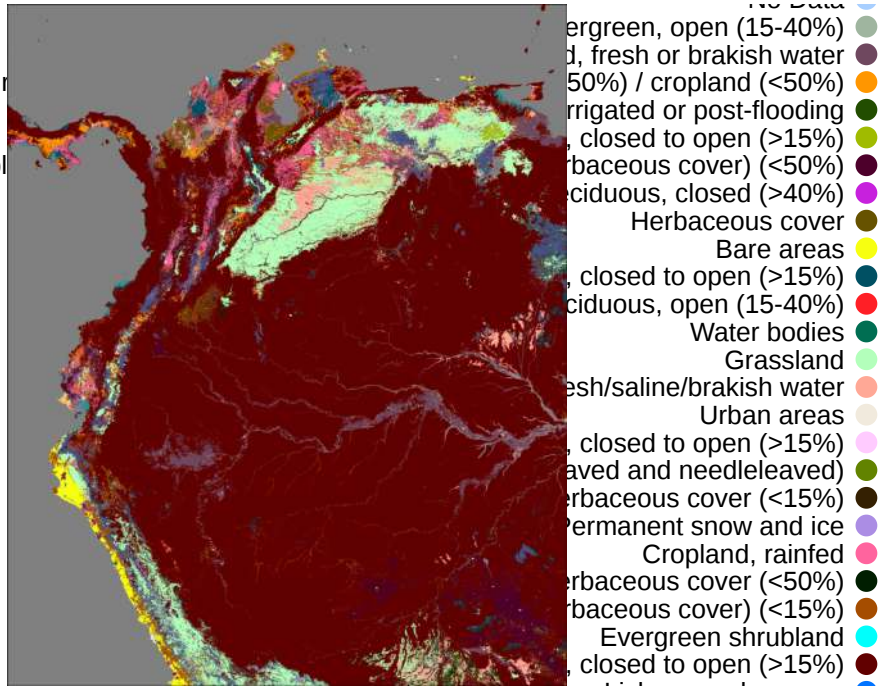



In [22]:
```
# Load land cover legend as dictionary
clcdic = include(string(pathin2,"/ESA_LC/legend/ESALClegend2014.jl"));
```

In [23]:
```
# Assign dictionary as propertes of the clc cube
clc.properties["labels"]=include(string(pathin2,"/ESA_LC/legend/ESALClegend2014.jl"));
```

In [24]:
```
plotMAP(clc)
```

Out[24]:




## Example of mask for selected land cover types and extraction of PCs information

In [25]:
```
# Get unique values of land cover types in the study area
clcid = unique(clc[:,:]);
```

In [26]:
```
i = 2
clcdic[clcid[i]]
```

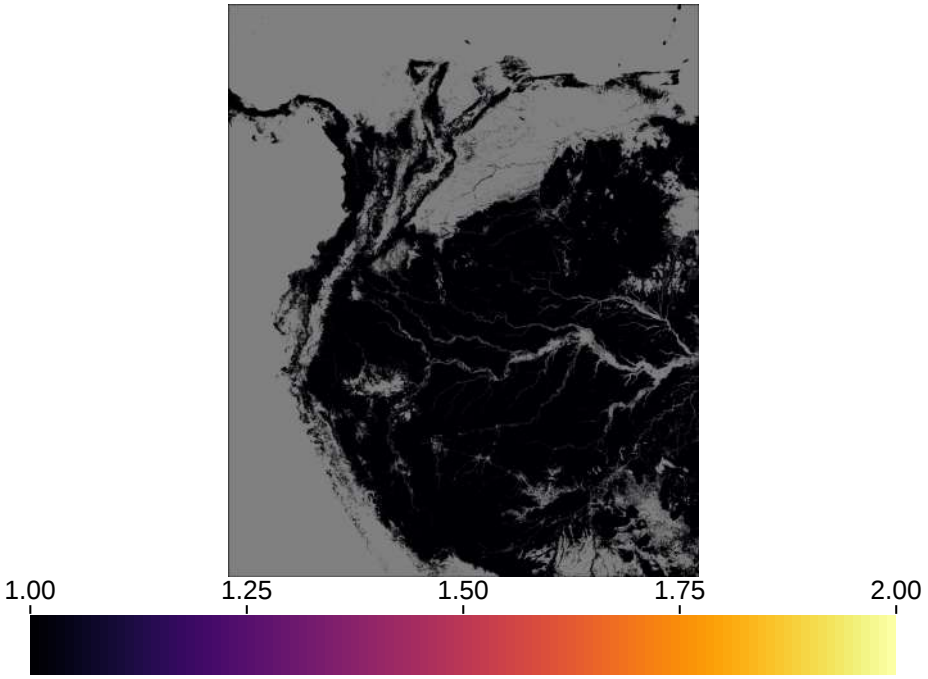
Out[26]:
```
"Tree cover, broadleaved, evergreen, closed to open (>15%)"
```

In [27]:
```
i=2
clcmask = map(x-> ismissing(x) ? missing : x==clcid[i] ? 1 : missing, clc) #
[lon=lon,lat=lat]
```

Out[27]:
```
Transformed cube In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 44.22 MB
```

In [28]:
```
plotMAP(clcmask)
```

Out[28]:




In [29]:
```
pcasig[Components_or_Var="sigPC_var1"]
```

Out[29]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 44.22 MB
```

In [30]:
```
clcmask
```

Out[30]:
```
Transformed cube In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 44.22 MB
```

In [31]:
```
commask1 = map((x,y)->x*y, pcasig[Components_or_Var="sigPC_var1"], clcmask)
commask2 = map((x,y)->x*y, pcasig[Components_or_Var="sigPC_var2"], clcmask)
commask3 = map((x,y)->x*y, pcasig[Components_or_Var="sigPC_var3"], clcmask)
```

Out[31]:
```
Transformed cube ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 44.22 MB
```

In [32]:
```
heatmap(commask1[:,:]'[end:-1:1,:], aspect_ratio = :equal)#, dmin=0)
```

Out[32]:

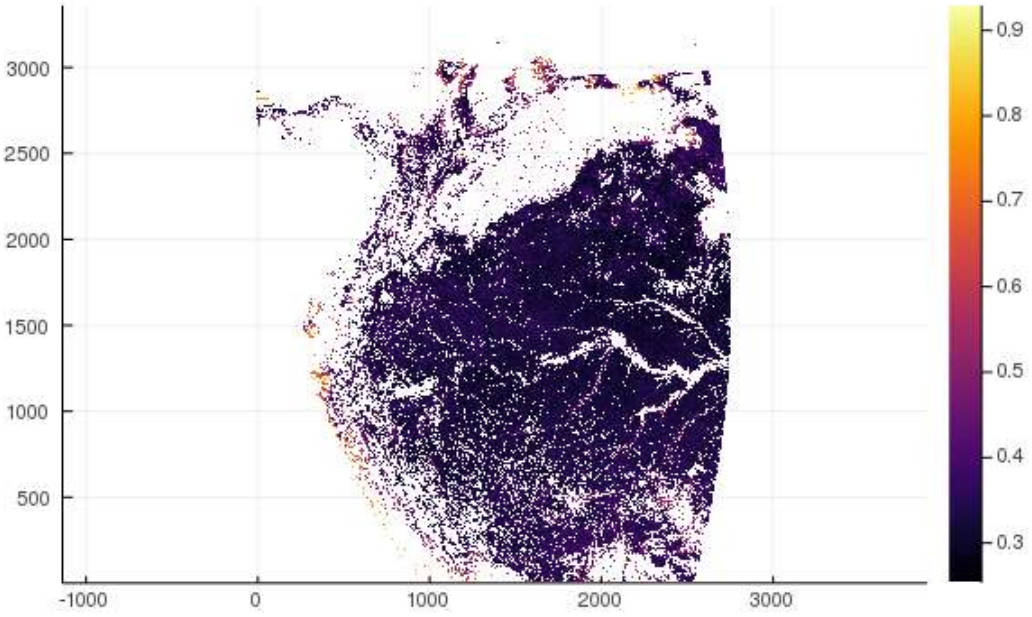



**Figure 5: Histogram of variance explained by the first three principal components**

### Define function for plotting histograms

In [33]:
```
function plotPCshisto(i, clcid, clc, pcasig, titlen; legendYN = :false)
# titlen = clcdic[clcid[i]]

# Generate mask for the selected land cover type
clcmask = map(x-> ismissing(x) ? missing : x==clcid[i] ? 1 : missing, clc)

# Select data of each PCs based on the land cover type (mask generation)
@show(i)
commask1 = map((x,y)->x*y, pcasig[Components_or_Var="sigPC_var1"], clcmask)
commask2 = map((x,y)->x*y, pcasig[Components_or_Var="sigPC_var2"], clcmask)
commask3 = map((x,y)->x*y, pcasig[Components_or_Var="sigPC_var3"], clcmask)

# Filter data that is higher than zero
h1 = filter(x->x>0, skipmissing(commask1[:,:]))
h2 = filter(x->x>0, skipmissing(commask2[:,:]))
h3 = filter(x->x>0, skipmissing(commask3[:,:]));

varname = "Number of pixels"
xname = "Variance explained"

# Plot histograms
hbar = plot(h1, seriestype=:barhist, color="red", linealpha=0.0, label="Comp
onent 1",
    title = titlen, titlefont = font(10), legend = legendYN,
        xlabel = xname, xlim = (0,1), xtickfont = font(10),
        ylabel = varname, ytickfont=font(10), guidefont=font(10))
plot!(h2, seriestype=:barhist, color="green", linealpha=0.0, label="Componen
t 2")
plot!(h3, seriestype=:barhist,  color="blue", linealpha=0.0, label="Componen
t 3",
    size=(400,300))

return(hbar)

end
```

Out[33]:
```
plotPCshisto (generic function with 1 method)
```

## Broadleaf evergreen forest

In [34]:
```
i =2
titlen = clcdic[clcid[i]]
```

Out[34]:
```
"Tree cover, broadleaved, evergreen, closed to open (>15%)"
```

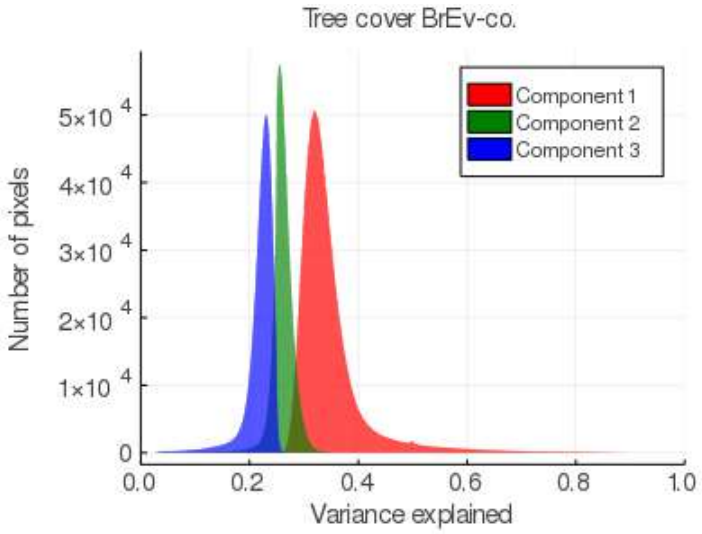
In [35]:
```
titlen = "Tree cover BrEv-co."
```

Out[35]:
```
"Tree cover BrEv-co."
```

In [36]:
```
# Replace title with a shorter name
plotPCshisto(i, clcid, clc, pcasig, titlen; legendYN = :true)
```

```
i = 2
```

Out[36]:




## Shrubland

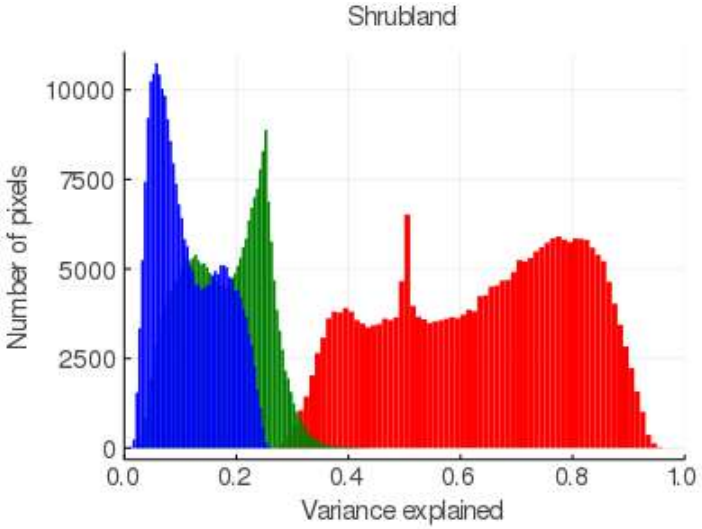
In [37]:
```
i = 15 # Shrubland
titlen = clcdic[clcid[i]]
```

Out[37]:
```
"Shrubland"
```

In [38]:
```
plotPCshisto(i, clcid, clc, pcasig, titlen; legendYN = :false)
```

```
i = 15
```

Out[38]:




## Grassland

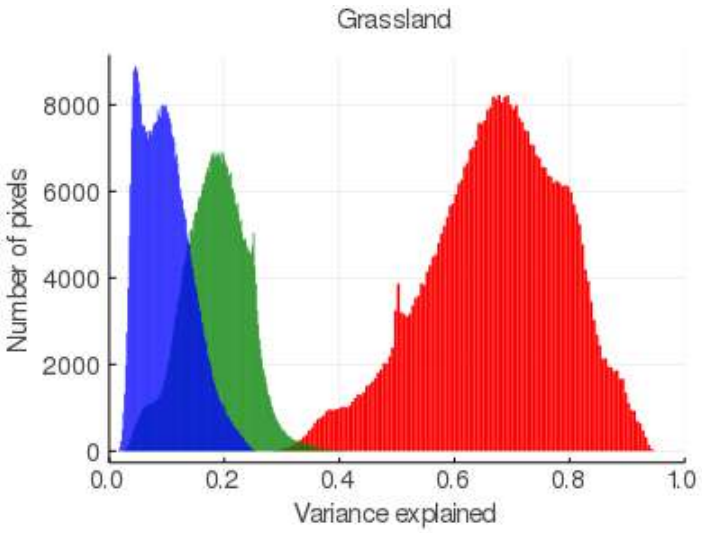
In [39]:
```
i = 14 # Grassland
titlen = clcdic[clcid[i]]
```

Out[39]:
```
"Grassland"
```

In [40]:
```
plotPCshisto(i, clcid, clc, pcasig, titlen; legendYN = :false)
```

```
i = 14
```

Out[40]:




## Herbaceous

In [41]:
```
i =12
titlen = clcdic[clcid[i]]
```

Out[41]:
```
"Mosaic herbaceous cover (>50%) / tree and shrub (<50%)"
```

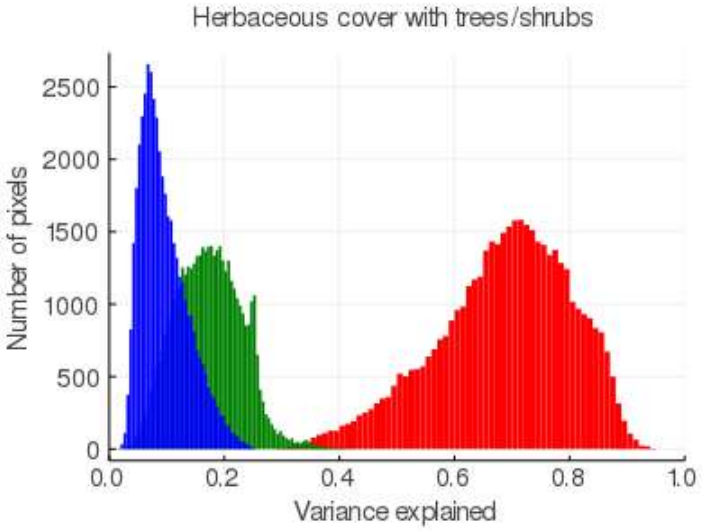
In [44]:
```
# Replace title with a shorter name
titlen = "Herbaceous cover with trees/shrubs"#
```

Out[44]:
```
"Herbaceous cover with trees/shrubs"
```

In [45]:
```
plotPCshisto(i, clcid, clc, pcasig, titlen; legendYN = :false)
```

```
i = 12
```

Out[45]:



# Main figures

# Figure 6: (6B) Seasonality ratio map of annual and semiannual oscillations from the Mean Seasonal Cycle (MSC) and (6A) plots from three pixels

### Estupinan-Suarez, et al. (2021). A Regional Earth System Data Lab for Understanding Ecosystem Dynamics: An Example from Tropical South America. Front. Earth Sci. 9:613395. doi: 10.3389/feart.2021.613395

**Correspondence to: lestup@bgc-jena.mpg.de, linamaesu@gmail.com**

**GitHub repository: https://github.com/linamaes/Regional_ESDL (https://github.com/linamaes/Regional_ESDL)**

This script does the following:

- Computes the Mean Seasonal Cycle (MSC) of the first principal component from the PCA analysis. For more details about PCA analysis see script 03 - PCA processing.
- Imports and applies a mask based on MODIS quality flags. See the mask generation script 01.
- Calcultes the ratio between annual and semiannual oscillation based on the Fast Fourier Spectrum.

About the notebook

- It is written in Julia 1.3
- "#" comments in the code are intended to explain specific aspects of the coding
- New steps in workflows are introduced with bold headers

March 2021, Max Planck Institute for Biogeochemistry, Jena, Germany

# Load Packages

In [1]:
```
using ESDL
```

In [2]:
```
using ESDLPlots
```

**Unable to load WebIO. Please make sure WebIO works for your Jupyter client. For troubleshooting, please see the WebIO/IJulia documentation (https://juliagizmos.github.io/WebIO.jl/latest/providers/ijulia/).**

In [3]:
```
using Plots
```

In [4]:
```
using FFTW
```

In [5]:
```
using Distributed
```

In [6]:
```
using CSV
```

In [7]:
```
using DelimitedFiles
```

In [8]:
```
gr(size=(600,400))
default(fmt = :png)
```

## Paths for loading and saving files

In [9]:
```
pathmask = ".../my_pathin/"
pathout = ".../my_pathout/"
```

Out[9]:
```
".../my_pathout/"
```

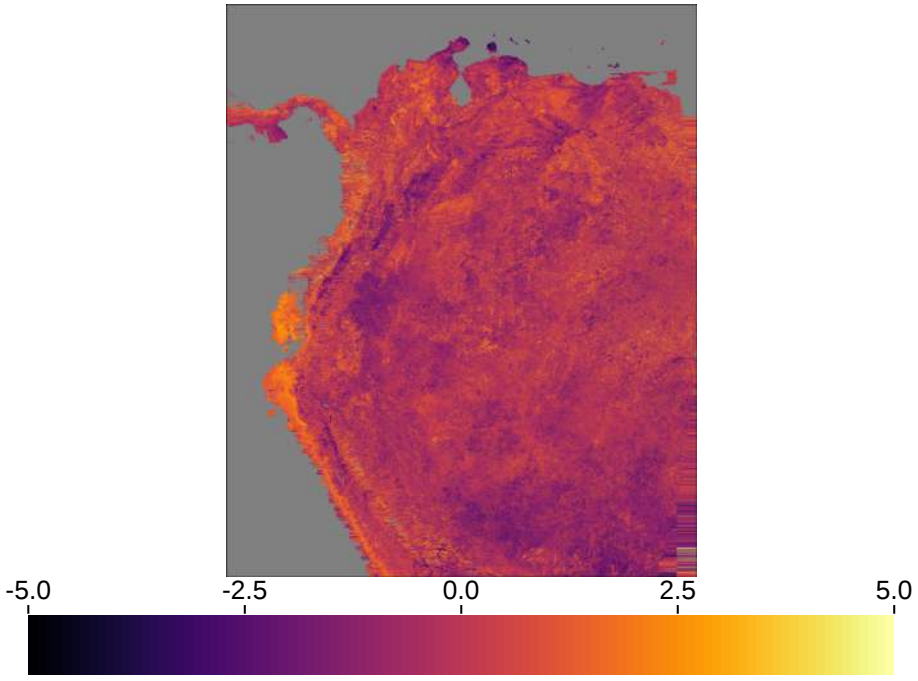
## Load data

In [11]:
```
pcacom = loadCube("pcacomstdTS1pc1km2014_mask") # Component 1
```

Out[11]:
```
ZArray Cube with the following dimensions
Time                 Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 27.81 GB
```

In [12]:
```
plotMAP(pcacom, time=Date(2007,12,31), dmin=-5, dmax=5)
```

Out[12]:



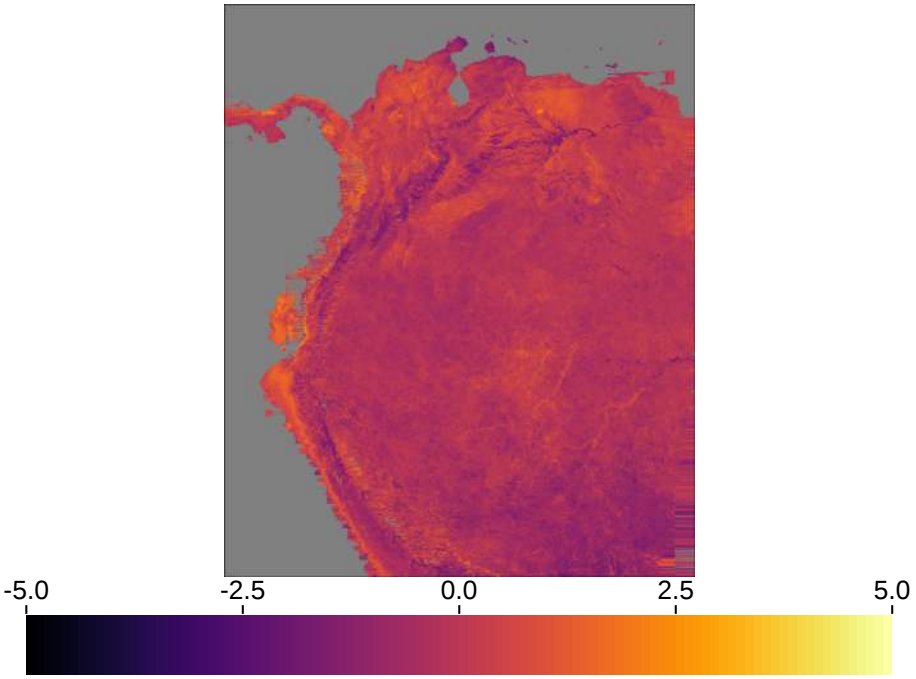

In [13]:
```
cmsc = getMSC(pcacom)
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:00:49
```

Out[13]:
```
ZArray Cube with the following dimensions
MSC                  Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 1.99 GB
```

In [14]:
```
plotMAP(cmsc, MSC=Date(1900,1,1), dmin=-5, dmax=5)
```

Out[14]:




## Load mask

In [15]:
```
# Load mask
maskin = CSV.read(string(pathmask, "msc_missings_mask.csv"), header=false);
```

In [16]:
```
# Convert NaNs to missings and transpose matrix
mask = map(x-> isnan(x) ? missing : x==0 ? missing : x, Matrix(maskin));
mask = mask' |> Array;
```

## Apply mask

In [17]:
```
function maskout(xout, xin, mask)
    xout[:] = xin.*mask
end
```

Out[17]:
```
maskout (generic function with 1 method)
```

In [18]:
```
indims = InDims("Lon","Lat")
outdims = OutDims("Lon","Lat")
```

Out[18]:
```
OutDims((ESDL.Cubes.Axes.ByName("Lon"), ESDL.Cubes.Axes.ByName("Lat")), (),
zero, identity, :auto, false, AsArray(), :input, Zarr.NoCompressor(), "", fa
lse, 1)
```

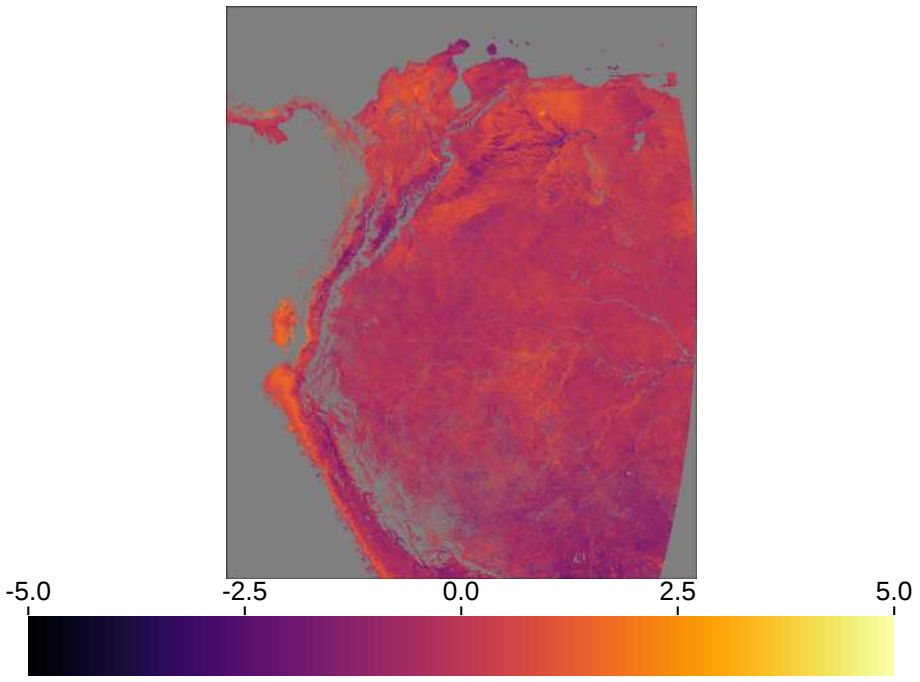
In [19]:
```
cmscm = mapCube(maskout, cmsc, mask, indims=indims, outdims=outdims)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:00:41
```

Out[19]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
MSC                 Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Total size: 1.99 GB
```

In [20]:
```
plotMAP(cmscm, MSC=Date(1900,1,1), dmin=-5, dmax=5)
```

Out[20]:




## Figure 6B - Ratio map between annual and semiannual oscillations

In [21]:
```
# This fuction computes the ratio between 2nd and (3rd+4th) fft frequency po
wer
# from fftw (fft[2]/(fft[3]+fft[4])

function spratioFx2(xin)
    any(ismissing, xin) && return missing
    xfft = fft(convert(Vector{Float64}, xin))
    xout = abs(xfft[2])/(abs(xfft[3])+abs(xfft[4]))
end
```

Out[21]:
```
spratioFx2 (generic function with 1 method)
```

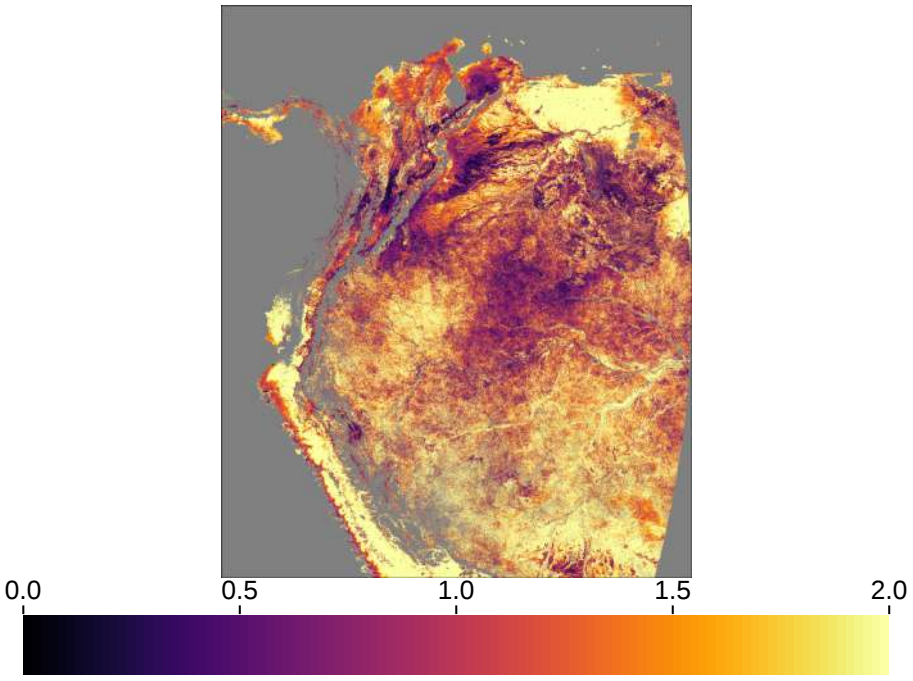
In [22]:
```
ratiosp2 = mapslices(spratioFx2, cmscm, dims="MSC")
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:05:40
```

Out[22]:
```
In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 44.22 MB
```

In [23]:
```
# Colors differ with the publication colorscheme. The publication map was done using matplotlib in Python
plotMAP(ratiosp2, dmin=0, dmax=2)
```

Out[23]:




In [24]:
```
ratioout2 = map(x -> ismissing(x) ? NaN : x, (ratiosp2[:,:]' |> Matrix));
# writedlm(string(pathout,"ratio_annual_semiannual_maskosc234.csv"), ratioout2, ",")
```

## Percentage annual and semiannual oscillations

In [25]:
```
function perAnnSemFx(xin)
    any(ismissing, xin) && return missing
    xfft = fft(convert(Vector{Float64}, xin))
    absx = map(x->abs(x), xfft)
    xout = (abs(xfft[2])+abs(xfft[3])+abs(xfft[4]))/sum(absx[2:23])
    return xout
end
```

Out[25]:
```
perAnnSemFx (generic function with 1 method)
```

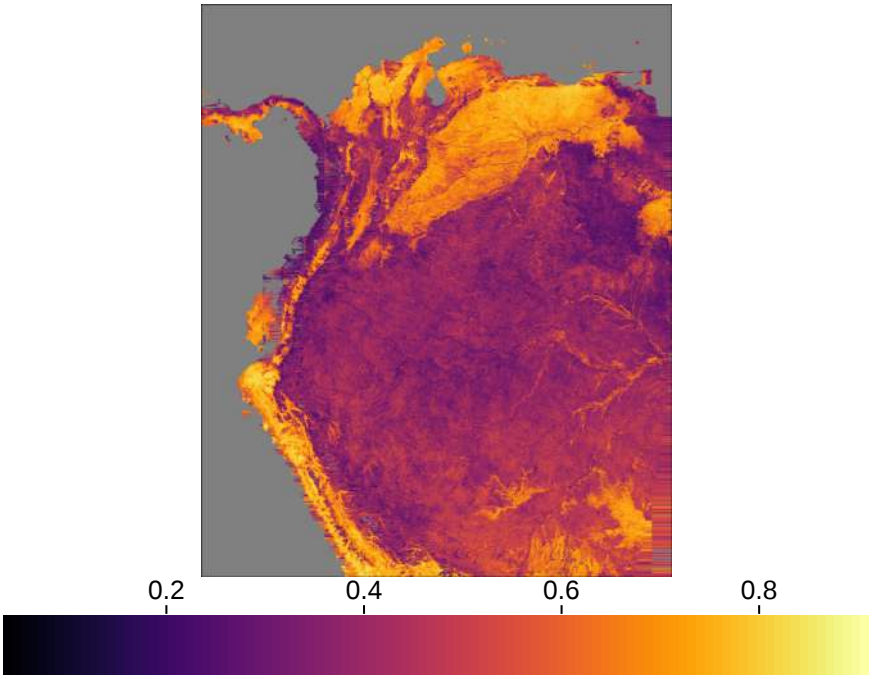
In [26]:
```
percentannsem = mapslices(perAnnSemFx, cmsc, dims="MSC")
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:06:33
```

Out[26]:
```
In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 44.22 MB
```

In [27]:
```
plotMAP(percentannsem)
```

Out[27]:




## Figure 6A - MSC plots by pixels

### TS decomposition using Fast Fourier

In [28]:
```
addprocs(8);
```

In [29]:
```
@everywhere using ESDL
```

In [30]:
```
cfft = filterTSFFT(pcacom)
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:04:16m
```

Out[30]:
```
ZArray Cube with the following dimensions
Time                 Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Scale                Axis with 4 elements: Trend Long-Term Variability Annual
Cycle Fast Oscillations
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 111.24 GB
```

In [31]:
```
cfftmsc = getMSC(cfft[Scale="Annual Cycle"])
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:00:24
```

Out[31]:
```
ZArray Cube with the following dimensions
MSC                  Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 1.99 GB
```

In [32]:
```
rmprocs(workers())
```

Out[32]:
```
Task (done) @0x00007f29d9c2fd00
```

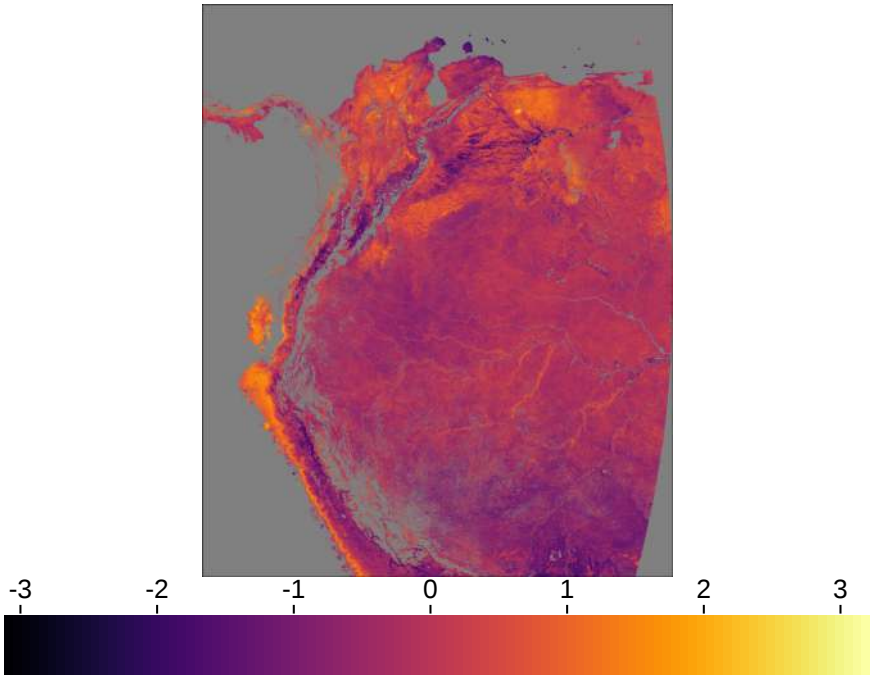
In [33]:
```
cfftmscm = mapCube(maskout, cfftmsc, mask, indims=indims, outdims=outdims)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:00:44
```

Out[33]:
```
ZArray Cube with the following dimensions
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
MSC                  Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Total size: 1.99 GB
```

In [34]:
```
plotMAP(cfftmscm, MSC=Date(1900,1,1))
```

Out[34]:




## Calculate mean seasonal standard deviation of PC1 TS

In [35]:
```
addprocs(6);
```

In [36]:
```
pcacom
```

Out[36]:
```
ZArray Cube with the following dimensions
Time                 Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 27.81 GB
```

In [37]:
```
@everywhere function getsdSC(xout, xin, timeax)
    any(ismissing, xin) && return missing
    timedoy = map(x->Dates.dayofyear(x), timeax)
    #xout = zeros(46)
    for i in 1:46
        idx = findall(x-> x==timedoy[i], timedoy)
        xout[i] = std(map(x->xin[x], idx))
    end
    return xout
end
```

In [38]:
```
axmsc = cmsc.axes[1]
```

Out[38]:
```
MSC                 Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
```

In [39]:
```
indimssdsc = InDims("Time")
outdimssdsc = OutDims(axmsc)
```

Out[39]:
```
OutDims((ESDL.Cubes.Axes.ByValue(MSC                 Axis with 46 Elements f
rom 1900-01-01T00:00:00 to 1900-12-27T00:00:00),), (), zero, identity, :aut
o, false, AsArray(), :input, Zarr.NoCompressor(), "", false, 1)
```

In [40]:
```
@everywhere using Dates, OnlineStats
```

In [41]:
```
tall = collect(pcacom.axes[1].values);
```

In [42]:
```
csdsc = mapCube(getsdSC, pcacom, tall, indims=indimssdsc, outdims=outdimssds
c)
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:01:08
```

Out[42]:
```
ZArray Cube with the following dimensions
MSC                 Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 1.99 GB
```

In [43]:
```
csdscfft = mapCube(getsdSC, cfft[Scale="Annual Cycle"], tall, indims=indimss
dsc, outdims=outdimssdsc)
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:01:08
```

Out[43]:
```
ZArray Cube with the following dimensions
MSC                 Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 1.99 GB
```

In [44]:
```
rmprocs(workers())
```

Out[44]:
```
Task (done) @0x00007f29ab2735b0
```

In [45]:
```
csdscm = mapCube(maskout, csdsc, mask, indims=indims, outdims=outdims)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:00:44
```

Out[45]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
MSC                 Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Total size: 1.99 GB
```

In [46]:
```
csdscfftm = mapCube(maskout, csdscfft, mask, indims=indims, outdims=outdims)
```

```
┌ Warning: There are still cache misses
└ @ ESDL.DAT /Net/Groups/BGI/scratch/lestup/julia_atacama_depots/packages/ES
DL/skMpG/src/DAT/DAT.jl:608
Progress: 100%|████████████████████████████████████████| Time: 0:00:42
```

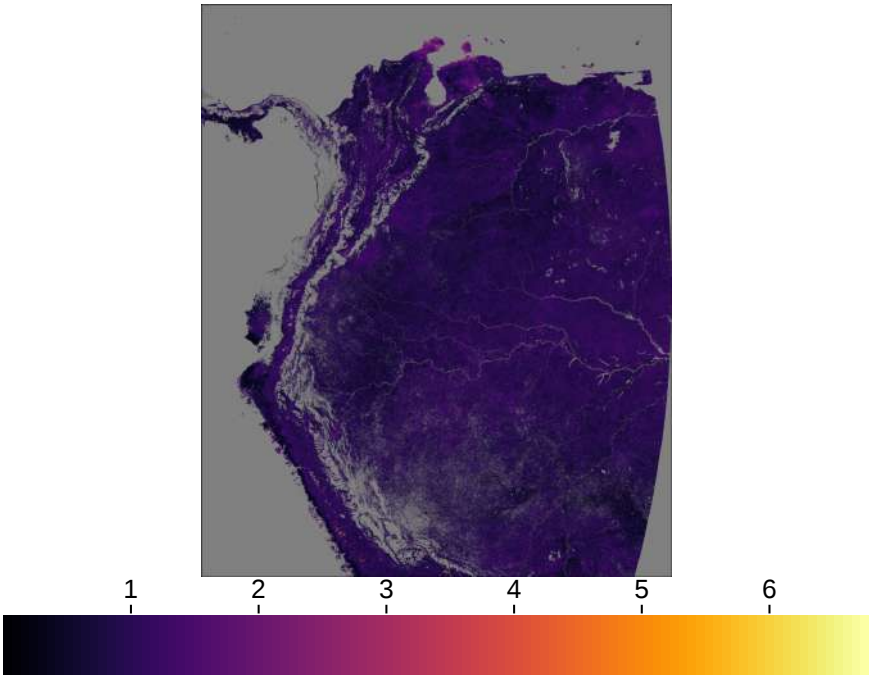
Out[46]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
MSC                 Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Total size: 1.99 GB
```

In [47]:
```
workers()
```

Out[47]:
```
1-element Array{Int64,1}:
 1
```

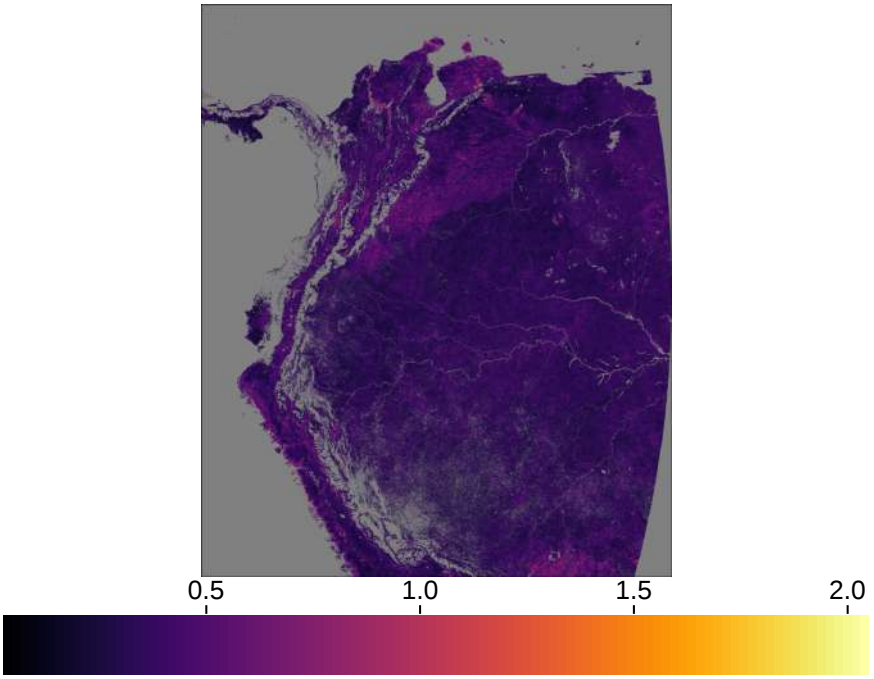
In [48]:
```
plotMAP(csdscm, MSC=Date(1900,1,1))
```

Out[48]:



In [49]:
```
plotMAP(csdscfftm, MSC=Date(1900,12, 27))
```

Out[49]:




## Save cubes

In [25]:
```
# saveCube(ratiosp2, "ratio_annual_semiannual_pca1kmstd2014_mask")
```

In [50]:
```
# saveCube(cmscm, "pcacom1msc1km2014_mask")
```

In [51]:
```
# saveCube(cfftmscm, "pcafftmsc1kmstd2014_mask")
```

In [52]:
```
# saveCube(csdscm, "pcacom1sdsc1km2014_mask")
```

In [53]:
```
# saveCube(csdscfftm, "pcafftsdsc1kmstd2014_mask")
```

## Plot MSC for some pixels

In [54]:
```
# Get the longitude and latitude coordinates
lonaxc = collect(cmsc.axes[2].values)
lataxc = collect(cmsc.axes[3].values);
```

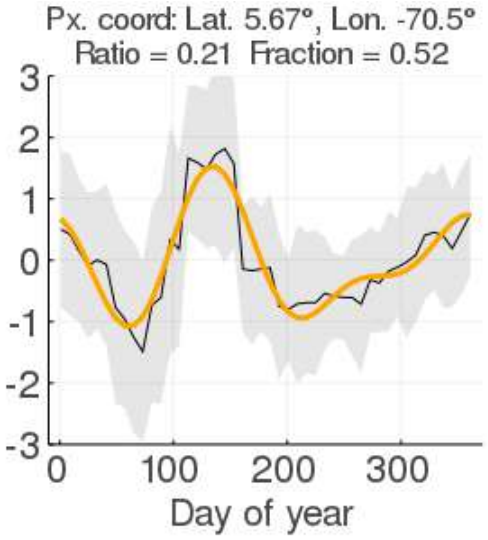
In [55]:
```
# Define days of the year from MSC
taxin = collect(cmsc.axes[1].values);
tax = map(x->Dates.dayofyear(x), taxin);
```

In [56]:
```
# foldp = "plots/pxs"

# Define function for plotting

function plotpx(clon, clat)

    titlen = "\nRatio = "#"\nRatio (ann/sem) = "
    titlen2 = "  Fraction = " #"\nPercentage (ann&sem) = "


    coordout = string("Px. coord: Lat. ", round(lataxc[clat], digits=2) ,"°, Lon. ", round(lonaxc[clon], digits=2),"°",
        titlen, round(ratiosp2[clon,clat], digits=2),
        titlen2, round(percentannsem[clon,clat], digits=2))

    p1 = plot(tax, cmsc[:,clon, clat].*-1, label  = "PC1 MSC", color="black",
        ribbon=csdsc[:,clon,clat], fillalpha=0.1, lw=1,
        xlabel="Day of year", xguidefontsize=14, xtickfontsize=15,
        ylim=(-3, 3), yguidefontsize=14,  ytickfontsize=15, #ylabel="PC1",
        title=coordout, titlefontsize = 12, legendfontsize = 7, legend=:false,
        size=(300, 300)) #ylim=(3, 8.52

    plot!(tax, cfftmsc[:, clon, clat].*-1, label = "FFT (annual)", lw=3, color="orange")

#     savefig(p1, string(rootp,foldp,"lon_",clon,"lat_",clat,"2014v2_mask",dpix,"dpi.png"))

    return p1
end
```

Out[56]:
```
plotpx (generic function with 1 method)
```

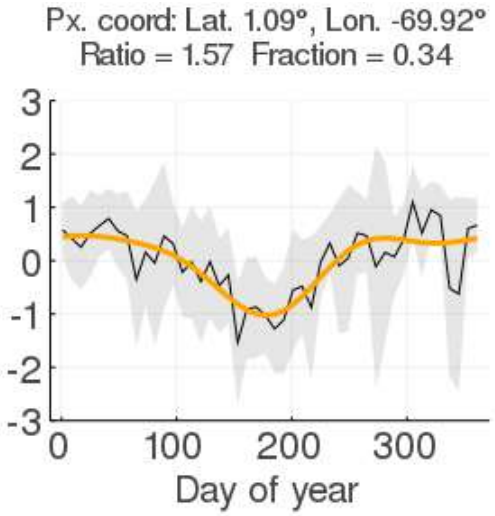
In [57]:
```
px1 = plotpx(1500, 1000)
```

Out[57]:



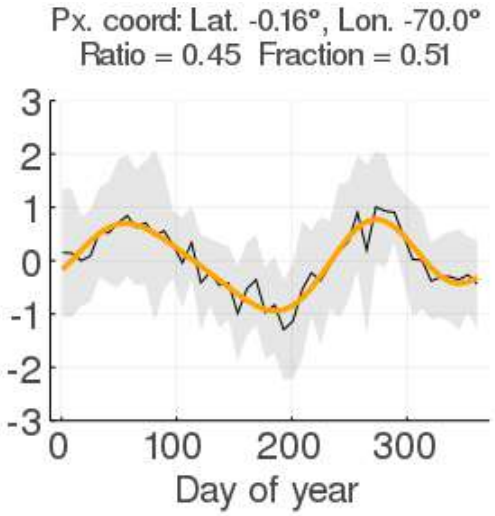
In [58]:
```
px2 = plotpx(1570, 1550)
```

Out[58]:



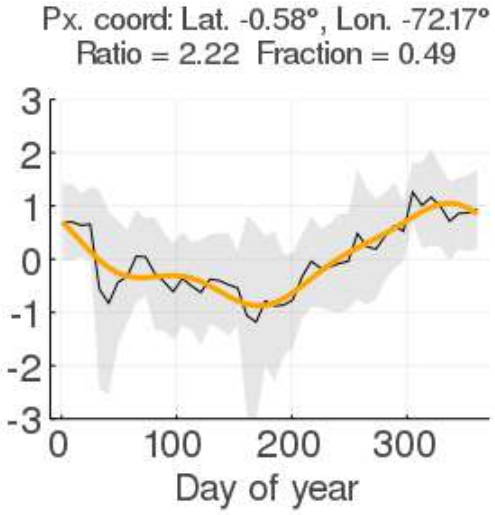

In [59]:
```
px3 = plotpx(1560, 1700)
```

Out[59]:




In [60]:
```
px4 = plotpx(1300,1750)
```

Out[60]:



# Main figures

# Figure 7: Plots of the Mean Seasonal Cycle (MSC) of the first principal component by biotic units (bu)

### Estupinan-Suarez, et al. (2021). A Regional Earth System Data Lab for Understanding Ecosystem Dynamics: An Example from Tropical South America. Front. Earth Sci. 9:613395. doi: 10.3389/feart.2021.613395

**Correspondence to: lestup@bgc-jena.mpg.de, linamaesu@gmail.com**

**GitHub repository: https://github.com/linamaes/Regional_ESDL (https://github.com/linamaes/Regional_ESDL)**

Steps:

- Load computed MSC (mean and std) for 1st PCA component
- Load ratio annual and semiannual ocsillation cube by bu
- Load percentage of variance explained by bu
- Plots MSC by biotic units. It corresponds to Figure 7A to 7D

About the notebook:

- It is written in Julia 1.3
- "#" comments in the code are intended to explain specific aspects of the coding
- New steps in workflows are introduced with bold headers

April 2021, Max Planck Insitute for Biogeochemistry, Jena, Germany

## Load packages ¶

In [1]:
```
using OnlineStats
```

In [2]:
```
using ESDL
```

In [3]:
```
using CSV
```

In [4]:
```
using Dates
```

In [5]:
```
using NPZ
```

In [6]:
```
using NetCDF
```

In [7]:
```
using Plots
```

In [8]:
```
using ESDLPlots
```

**Unable to load WebIO. Please make sure WebIO works for your Jupyter client. For troubleshooting, please see the WebIO/IJulia documentation (https://juliagizmos.github.io/WebIO.jl/latest/providers/ijulia/).**

In [9]:
```
using DelimitedFiles
```

In [10]:
```
gr(size=(600,400))
default(fmt = :png)
```

### Root

In [11]:
```
pathin = "/my_path_in/.../"
```

Out[11]:
```
"/path_in/.../"
```

## Load PCA 1st component (vegetation variables)

## Load cubes

In [13]:
```
percannsem = loadCube("fraction_biome_pca1km2014_mask")
```

Out[13]:
```
ZArray Cube with the following dimensions
Label1              Axis with 68 Elements from -3.4e38 to 67.0
Total size: 612.0 bytes
```

In [14]:
```
ratiosp = loadCube("ratio_biome_pca1km2014_mask")
```

Out[14]:
```
ZArray Cube with the following dimensions
Label1              Axis with 68 Elements from -3.4e38 to 67.0
Total size: 612.0 bytes
```

In [15]:
```
meanbybiome = loadCube("biomepcamsc1km2014_mask")
```

Out[15]:
```
ZArray Cube with the following dimensions
Label1              Axis with 68 Elements from -3.4e38 to 67.0
Category2           Axis with 46 Elements from -29219.0 to -28859.0
Total size: 27.49 KB
```

In [16]:
```
sdbybiome = loadCube("biomepcasdsc1km2014_mask")
```

Out[16]:
```
ZArray Cube with the following dimensions
Label1              Axis with 68 Elements from -3.4e38 to 67.0
Category2           Axis with 46 Elements from -29219.0 to -28859.0
Total size: 27.49 KB
```

In [17]:
```
cin = loadCube("pcacomstdTS1pc1km2014_mask")
```

Out[17]:
```
ZArray Cube with the following dimensions
Time                Axis with 644 Elements from 2001-01-05T00:00:00 to 2014-
12-31T00:00:00
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
Total size: 27.81 GB
```

## Load land cover by biotic units

In [18]:
```
lcbybuin = readdlm(string(pathin, "dataout/lc2014bybupercentage.csv"), ',');
```

In [19]:
```
# Convert missings from text to data missings
lcbybu = map(x->x=="missing" ? missing : x, lcbybuin)

# Exclude missings and NaNs
lcbybu2 = lcbybu[3:end,:];
```

## Get MSC dates axis

In [20]:
```
cin = getMSC(cin[lat=(4,6), lon=(-73,-71), time=2001:2005])
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:00:02
```

Out[20]:
```
In-Memory data cube with the following dimensions
MSC                 Axis with 46 Elements from 1900-01-01T00:00:00 to 1900-1
2-27T00:00:00
Lon                 Axis with 240 Elements from -72.99626135 to -71.00460265
Lat                 Axis with 240 Elements from 5.99616935 to 4.004510649999
999
Total size: 12.63 MB
```

In [21]:
```
taxin = collect(cin.axes[1].values);
```

In [22]:
```
tax = map(x->Dates.dayofyear(x), taxin[1:46]);
```

# Plotting and save MSC by biotic untis

In [23]:
```
bunames = collect(ratiosp.axes[1].values);
```

In [24]:
```
meanbybiome
```

Out[24]:
```
ZArray Cube with the following dimensions
Label1              Axis with 68 Elements from -3.4e38 to 67.0
Category2           Axis with 46 Elements from -29219.0 to -28859.0
Total size: 27.49 KB
```

In [25]:
```
ratiosp
```

Out[25]:
```
ZArray Cube with the following dimensions
Label1              Axis with 68 Elements from -3.4e38 to 67.0
Total size: 612.0 bytes
```

In [26]:
```
# Dictionary without Spanish tilde
dict = include("bioticunits_name_notilde.jl");
dictlc2 = include("lc_legend_abr2.jl");
```

In [27]:
```
# Map dictionary to biotic units
buid = collect(meanbybiome.axes[1].values)
buname = map(x->dict[x], buid);
```

### Smoothing function

In [28]:
```
function getIndex_circ2(x::Vector,i::Integer)
while i<1 || i>length(x)
i = i<1 ? i+length(x) : i-length(x)
end
x[i]
end

function smoothMSC(xin;ns=2)
    any(ismissing,xin) && return(fill!(similar(xin),missing))
    scur = sum(getIndex_circ2(xin,i) for i=1-ns:1+ns)
    afac = 1/(2ns+1)
    xout = zeros(size(xin))
    for i=1:length(xin)
        xout[i]=scur*afac
        scur+=getIndex_circ2(xin,i+ns+1)-getIndex_circ2(xin,i-ns)
    end
    xout
end
```

Out[28]:
```
smoothMSC (generic function with 1 method)
```

In [29]:
```
# Smooth MSC from biotic unit for plotting purpose
smsc = mapslices(smoothMSC,meanbybiome,dims="Category2",ns=3)
```

Out[29]:
```
In-Memory data cube with the following dimensions
Category2           Axis with 46 Elements from -29219.0 to -28859.0
Label1              Axis with 68 Elements from -3.4e38 to 67.0
Total size: 27.49 KB
```

In [30]:
```
# Geta axis Biotic Units ID
mscbuax = collect(meanbybiome.axes[1].values);
```

## Plots for single biotic units

In [31]:
```
row = 2
```

Out[31]:
```
2
```

In [32]:
```
buloc = 6
buid = Integer(mscbuax[buloc])
```

Out[32]:
```
5
```

In [33]:
```
pathoutts = string(pathin,"/plots/bumscpcastd/mask/2014/axesequalsmoothLC/")
varname = "Seasonal oscillation (PCA comp. No. 1)"
nameout = "PCAvegstd"
buidx = 5
ylim1 = -3
ylim2 = 4
```

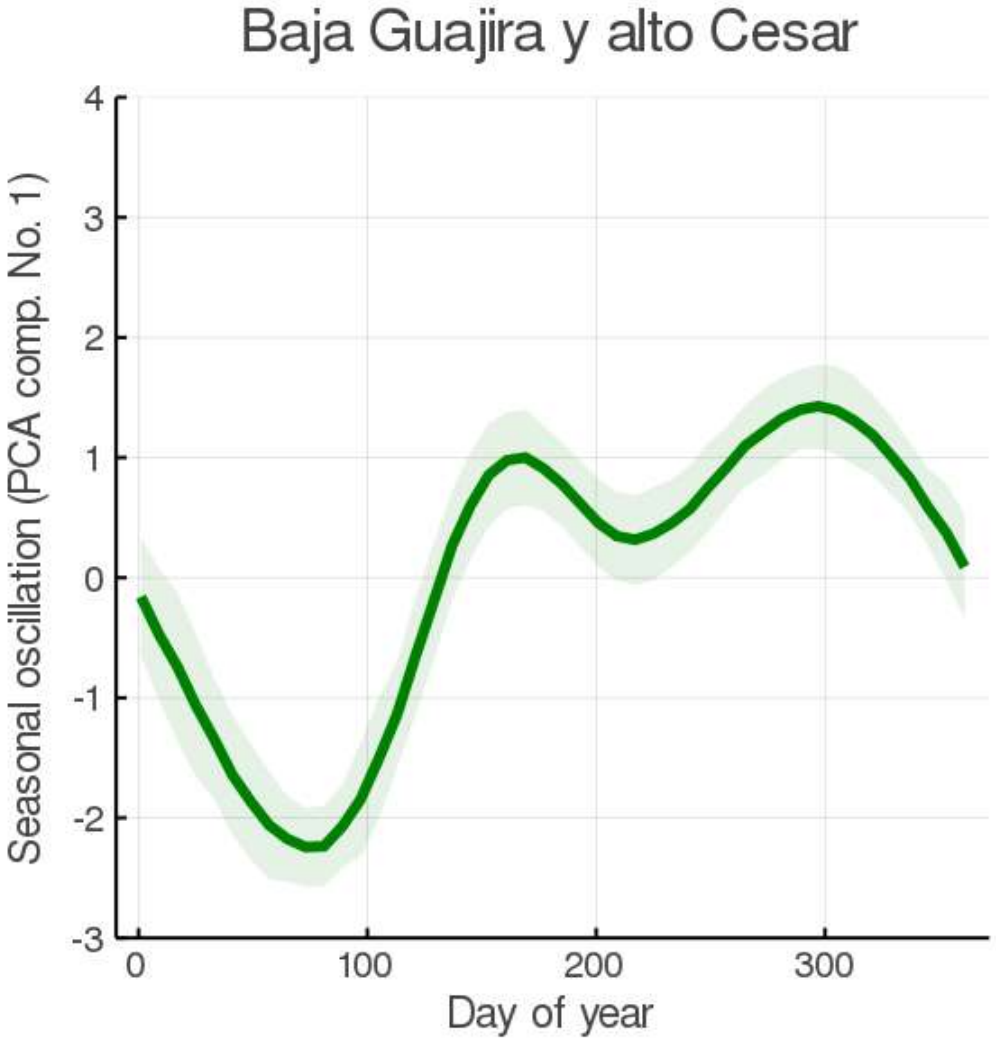
Out[33]:
```
4
```

In [34]:
```
titlen = dict[buid]
```

Out[34]:
```
"Baja Guajira y alto Cesar"
```

In [35]:
```
px = plot(tax, (smsc[:,buloc]*-1),
    color="green", legend=false,
    ribbon=sdbybiome[buloc,:], fillalpha=0.1,
    lw=3, xlabel="Day of year", xtickfont=font(7),
    ylabel=varname, ytickfont=font(7), guidefont=font(8),
    title=titlen, titlefontsize=11, legendfontsize=7,
    size=(300,300), ylim=(ylim1,ylim2), dpi=200)
```

Out[35]:




In [36]:
```
nameout2 = join(map(x -> isspace(titlen[x]) ? "" : titlen[x], 1:length(title
n)))
```

Out[36]:
```
"BajaGuajirayaltoCesar"
```

In [37]:
```
# Initialize px5 and px28
px5, px28 = 0, 0
```

Out[37]:
```
(0, 0)
```

## Looping figures

### Figures with dominant land cover names

In [38]:

```
for buloc = 2:size(ratiosp.axes[1])[1]

    buid = Integer(mscbuax[buloc])

#   Select the two most dominant land cover classes
    # Find  the row loction in lcbybu for selected by biotic unit "buidx"
    lcall = findall(row->row==buid, lcbybu2[:,1])
    # Filter land covers for the selected biotic unit
    lcsub = hcat(lcbybu2[lcall,2:end]' |> Matrix, lcbybu[1,2:end])
    # Sort land cover classes ascendingly including missings
    lcmis = sortslices(lcsub, dims=1, rev=true)
    # Exclude land cover classes with missing values
    idmis = findmax(findall(x->ismissing(x), lcmis[:,1]))[1]
    lcmax = lcmis[idmis+1:end,:]

      titlex = string(dictlc2[lcmax[1,2]], " (", round(lcmax[1,1], digits=2
), "%)
  & ", dictlc2[lcmax[2,2]], " (", round(lcmax[2,1], digits=2), "%)")

      coordout = string(titlex)

      titlen = dict[buid]

#   Smoothing function
     smsc = mapslices(smoothMSC, meanbybiome, dims="Category2",ns=3)

     px = plot(tax, meanbybiome[buloc,:].*-1, color="orange",
         lw=1.5, fillalpha=0.1, legend=:false)

#   Add values of the ratio and fraction to the plot
    annotate!([
            #(180,-4.3,text(titlen,7,:center,:gray)), # -> activate to add b
iotic units name on the bottom
           (300,3.2,text(string("\nRatio  = " , round(ratiosp[buloc], digits
=2),
            "\nFraction = ", round(percannsem[buloc], digits=2)),9))])

    plot!(tax, (smsc[:,buloc]*-1),
    color="green", legend=false,
    ribbon=sdbybiome[buloc,:], fillalpha=0.1,
    lw=3, xlabel="Day of year", xtickfont=font(7),
    ylabel=varname, ytickfont=font(7), guidefont=font(8),
    title=coordout, titlefontsize = 11, legendfontsize = 7,
    size=(300,300), ylim=(ylim1,ylim2), dpi=200)

#   Delete blank spaces from biotic units names
    nameout2 = join(map(x -> isspace(titlen[x]) ? "" : titlen[x], 1:length(t
itlen)))

#     savefig(px, string(pathoutts, nameout, "_bu_", buid, "_", nameout2, ".
png"))


#   Save two examples for demostrative purpose
#     buid == 28 ? px28=px : buid == 5 ? px5=px : print(0)
    if buid == 5
        px5=px
        println("example ", buid)
    elseif buid == 28
        px28=px
        println("example ", buid)
    end


end
```

```
example 5
example 28
```

In [39]:
```
titlen = dict[buid]
```

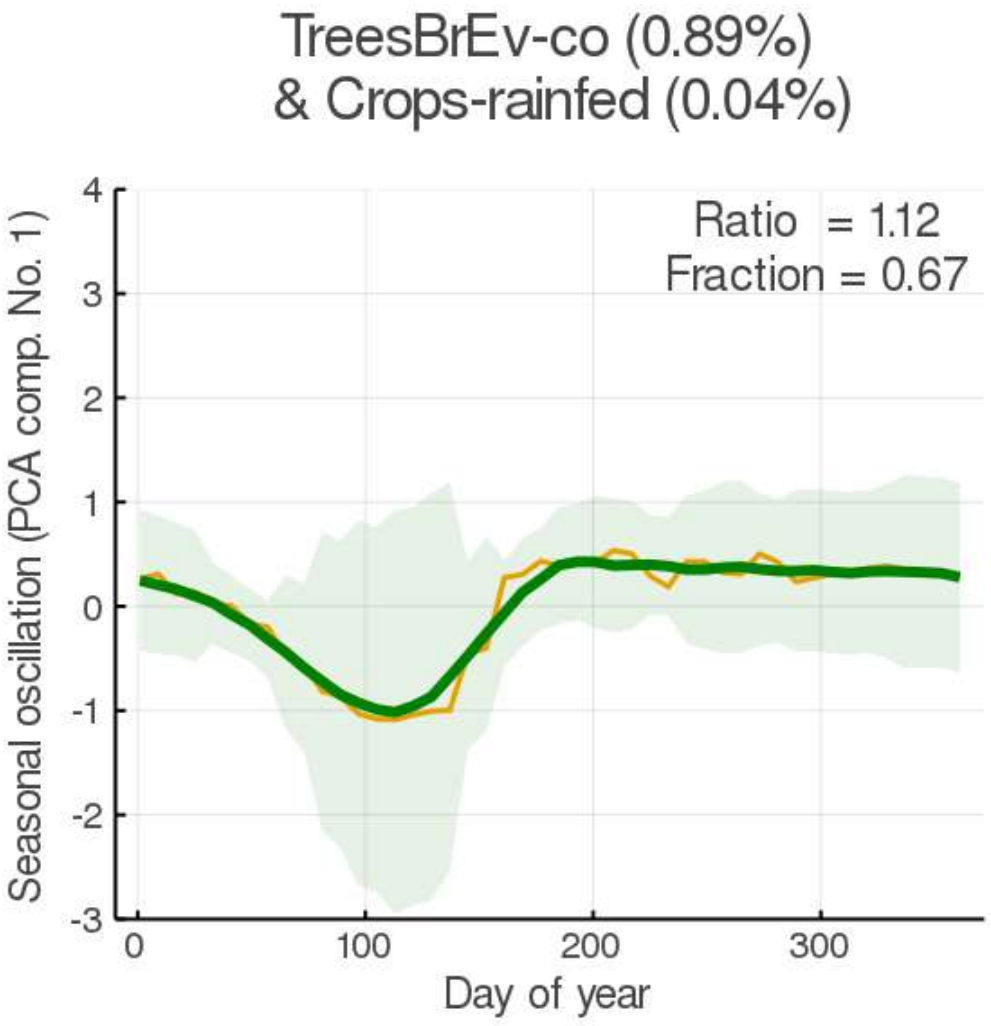
Out[39]:
```
"San Andres y Providencia"
```

In [40]:
```
buid, nameout2
```

Out[40]:
```
(67, "SanAndresyProvidencia")
```

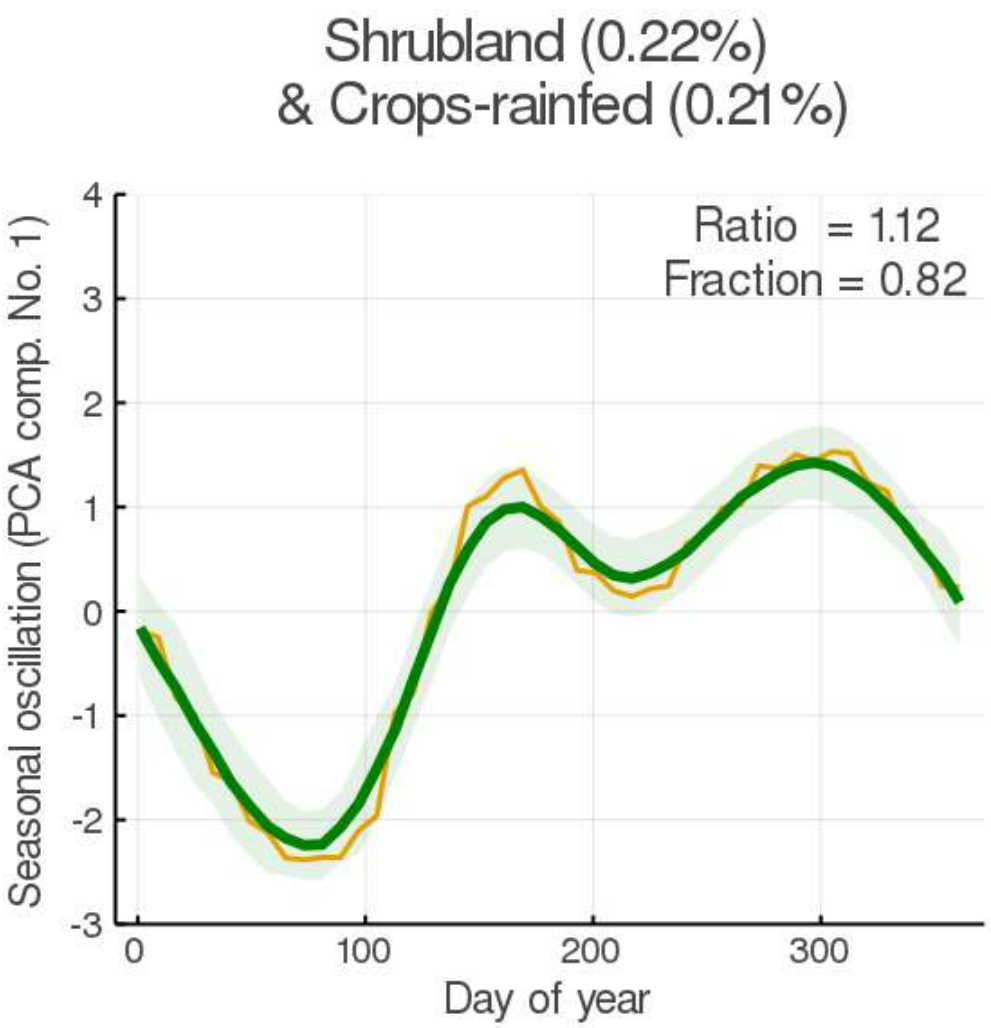
In [41]:
```
px
```

Out[41]:



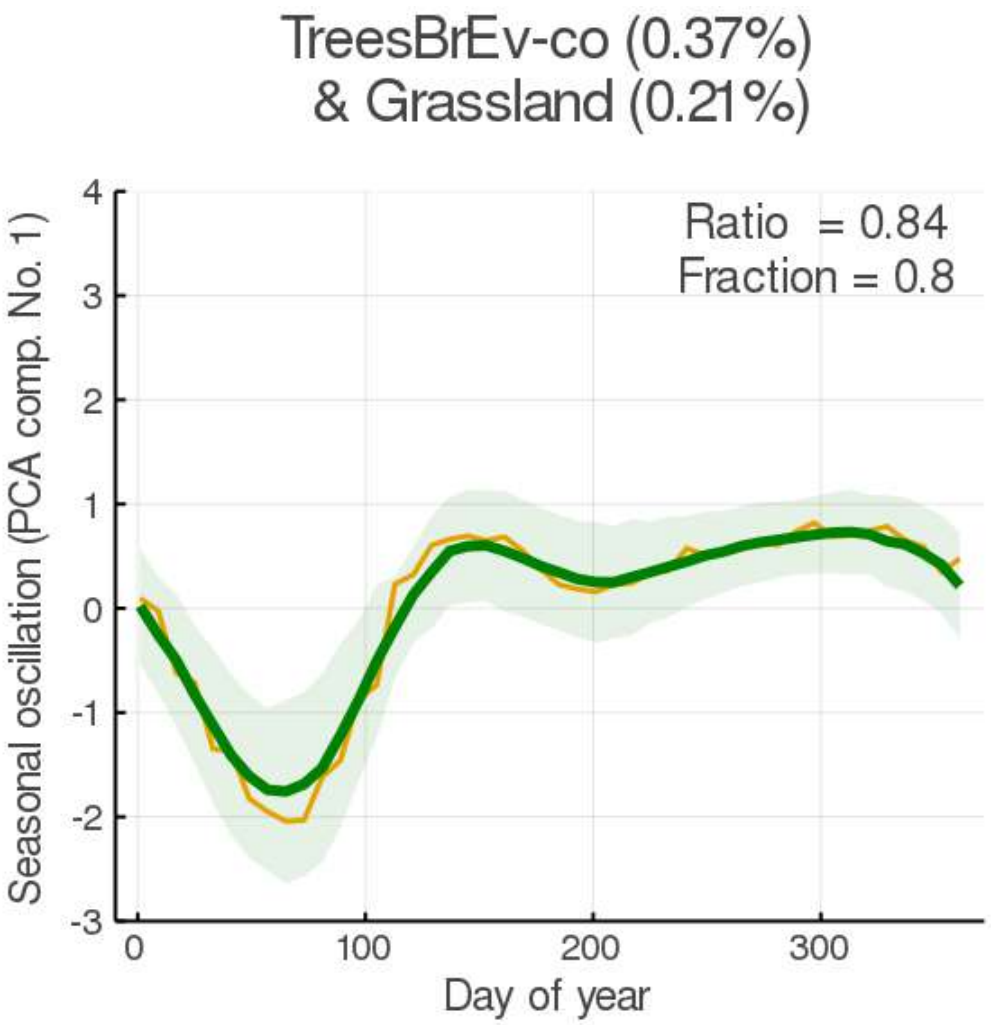
In [42]:
```
px5
```

Out[42]:



In [43]:
```
px28
```

Out[43]:



# Main and supplementary figures

# Figure 7, S4, S5: Heatmaps of seasonality ratio of annual/semiannual oscillations and climate variables

### Estupinan-Suarez, et al. (2021). A Regional Earth System Data Lab for Understanding Ecosystem Dynamics: An Example from Tropical South America. Front. Earth Sci. 9:613395. doi: 10.3389/feart.2021.613395

**Correspondence to: lestup@bgc-jena.mpg.de, linamaesu@gmail.com**

**GitHub repository: https://github.com/linamaes/Regional_ESDL (https://github.com/linamaes/Regional_ESDL)**

This script does the following:

- Loads cubes with seasonality ratio (annual/semiannual oscillation), biotic units map, land cover data by biotic units, and climate variables (precipitation of the driest month, maximum temperature of the warmest month, median annual cloud frecuency)
- Select climate data by biotic units
- Plot heatmaps of climate variables versus seasonality ratio

About the notebook:

- It is written in Julia 1.3
- "#" comments in the code are intended to explain specific aspects of the coding
- New steps in workflows are introduced with bold headers

April 2021, Max Planck Institute for Biogeochemistry, Jena, Germany

In [1]:
```
using OnlineStats
```

In [2]:
```
using ESDL
```

In [3]:
```
using ESDLPlots
```

**Unable to load WebIO. Please make sure WebIO works for your Jupyter client. For troubleshooting, please see the WebIO/IJulia documentation (https://juliagizmos.github.io/WebIO.jl/latest/providers/ijulia/).**

In [4]:
```
using Plots
```

In [5]:
```
using NetCDF
```

In [6]:
```
using DelimitedFiles
```

In [68]:
```
gr(size=(400,350)) # gr(size=(600,400)) #
default(fmt = :png)
```

# Load data

In [8]:
```
# Location RegESDL
pathin1 = "/my_pathin1/.../"

#Location land cover data by biotic units
pathin2 = "/my_pathin2/.../"

# Location Biotic units map in NetCDF format
pathin3 = "/my_pathin3/.../"
```

Out[8]:
```
"/my_pathin3/.../"
```

In [10]:
```
lat = (4,6)
lon = (-73,-71)
```

Out[10]:
```
(-73, -71)
```

In [11]:
```
time = (2001:2015)
```

Out[11]:
```
2001:2015
```

In [12]:
```
c = Cube(string(pathin1,"/Cube_2019highColombiaCube_184x120x120.zarr/"))
```

Out[12]:
```
Collection of ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Time                Axis with 782 Elements from 2001-01-05T00:00:00 to 2017-
12-31T00:00:00
Variable            Axis with 92 elements: S_Silt TemperatureSeasonality ..
MODCF_monthlymean_06 burn_date
Total size: 3.03 TB
```

In [13]:
```
# foreach(println,caxes(c)[4].values)
```

In [14]:
```
# Percentage of land cover by biotic untis
lcbybuin = readdlm(string(pathin2,"dimred/dataout/lc2014bybupercentage.csv"
), ',')
lcbybu = map(x->x=="missing" ? missing : x, lcbybuin)
lcbybu2 = lcbybu[3:end,:];
```

In [15]:
```
dictlc2 = include("lc_legend_abr2.jl");
```

## Defining mask from vegetation variables

In [16]:
```
cmask = c[variable="fapar", Time=Date(2001,1,5)]
```

Out[16]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 44.22 MB
```

In [17]:
```
mask1 = map(x->ismissing(x) ? missing : 1, cmask)
```

Out[17]:
```
Transformed cube ZArray Cube with the following dimensions
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 44.22 MB
```

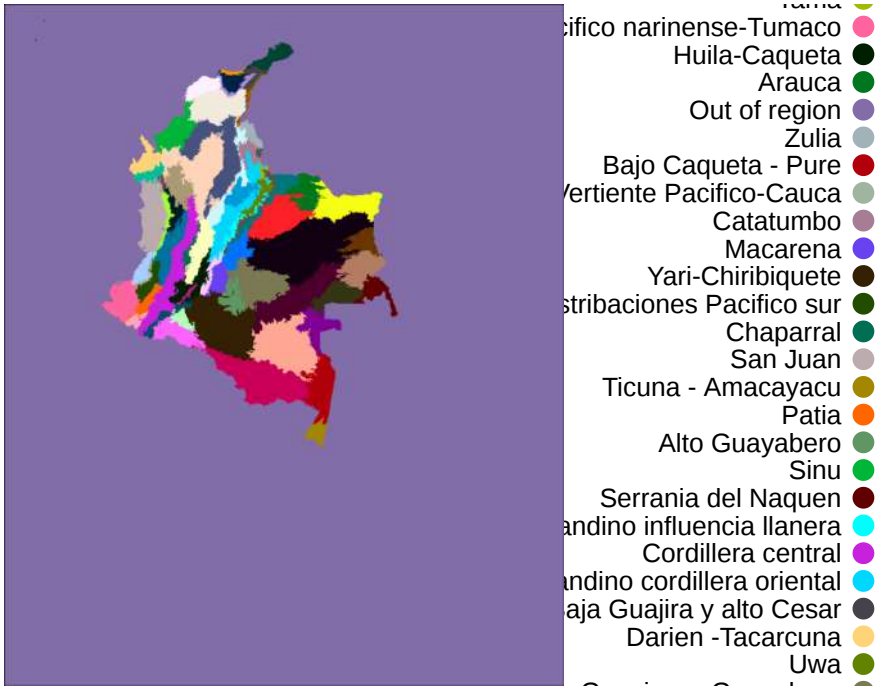
## Load biotic units from NetCDF

In [18]:
```
layer = ncread(string(pathin3,"/UB_IAvH_wgs84.nc"), "layer");
```

In [19]:
```
cbu = CubeMem(CubeAxis[getAxis("Lon", c),getAxis("Lat", c)], layer)
cbu.properties["labels"]=include("bioticunits_name_notilde.jl");
```

In [20]:
```
plotMAP(cbu, dmin=0, dmax=70)
```

Out[20]:




In [21]:
```
bunames = include("bioticunits_name_notilde.jl")
i = 1
bunames[i]
```

Out[21]:
```
"Alta Guajira"
```

## Load variables

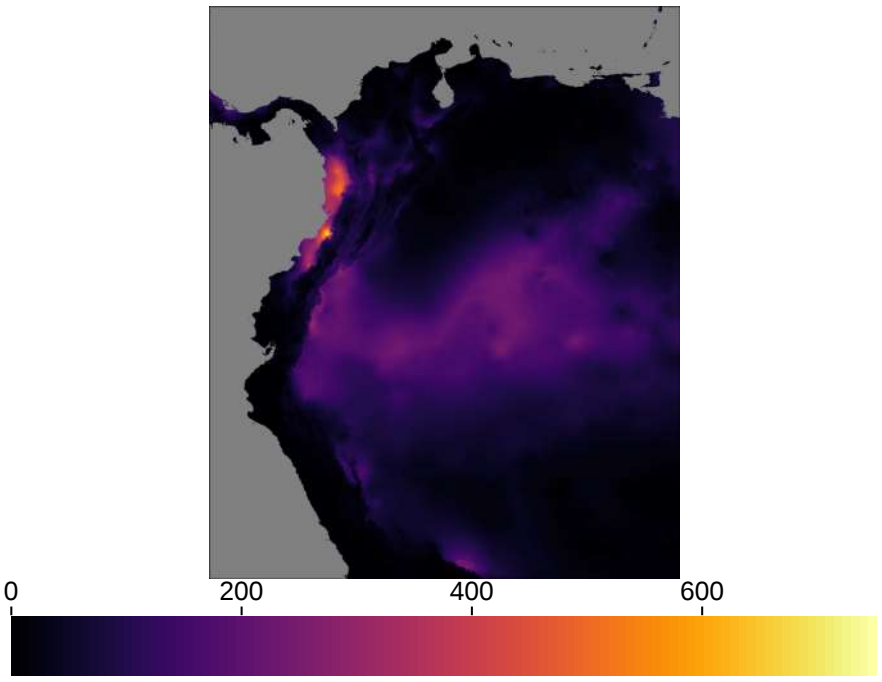
### Precipitation of the driest month

In [22]:
```
# This is a descriptive variable. It only has data on the first time-step
cppt = c[variable="PrecipitationofDriestMonth", Time=Date(2001,1,5)]
```

Out[22]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 44.22 MB
```

In [23]:
```
plotMAP(cppt)
```

Out[23]:



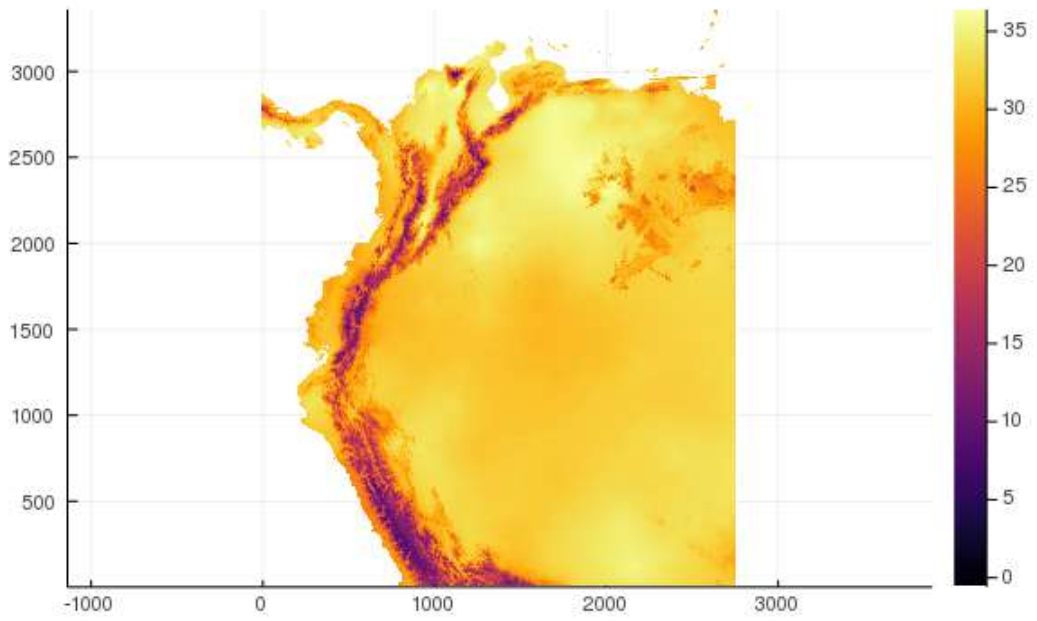

## Maximum temperature of the warmest month

In [24]:
```
ctem = c[variable="MaxTemperatureofWarmestMonth", Time=Date(2001,1,5)]
```

Out[24]:
```
ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 44.22 MB
```

In [25]:
```
heatmap(ctem[:,:]'[end:-1:1,:], aspect_ratio = :equal)
```

Out[25]:




## Mean monthly clouds data

In [26]:
```
ccloud = c[variable=["MODCF_monthlymean_01","MODCF_monthlymean_02","MODCF_monthlymean_03",
        "MODCF_monthlymean_04","MODCF_monthlymean_05","MODCF_monthlymean_06",
        "MODCF_monthlymean_07","MODCF_monthlymean_08","MODCF_monthlymean_09",
        "MODCF_monthlymean_11","MODCF_monthlymean_10","MODCF_monthlymean_12"],
    Time=Date(2001,1,5)]#, Time=Date(2001,1,5)]
```

Out[26]:
```
Collection of ZArray Cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.00464665
Lat                 Axis with 3360 Elements from 13.99613735 to -13.995417350000002
Variable            Axis with 12 elements: MODCF_monthlymean_01 MODCF_monthlymean_02 .. MODCF_monthlymean_10 MODCF_monthlymean_12
Total size: 530.64 MB
```

In [27]:
```
# Calculate annual median of clouds coverage
medianax = CategoricalAxis("Median", ["percentage"])
indims = InDims("Variable")
outdims = OutDims(medianax)
```

Out[27]:
```
OutDims((ESDL.Cubes.Axes.ByValue(Median                Axis with 1 elements: percentage ),), (), zero, identity, :auto, false, AsArray(), :input, Zarr.NoCompressor(), "", false, 1)
```

In [28]:
```
function medianFx(xout, xin)
    #@show size(xin)#, size(xin)
    sort!(xin)
    xout .= (xin[6]+xin[7])/200 # coefficient 2  * 100 to get %
end
```

Out[28]:
```
medianFx (generic function with 1 method)
```

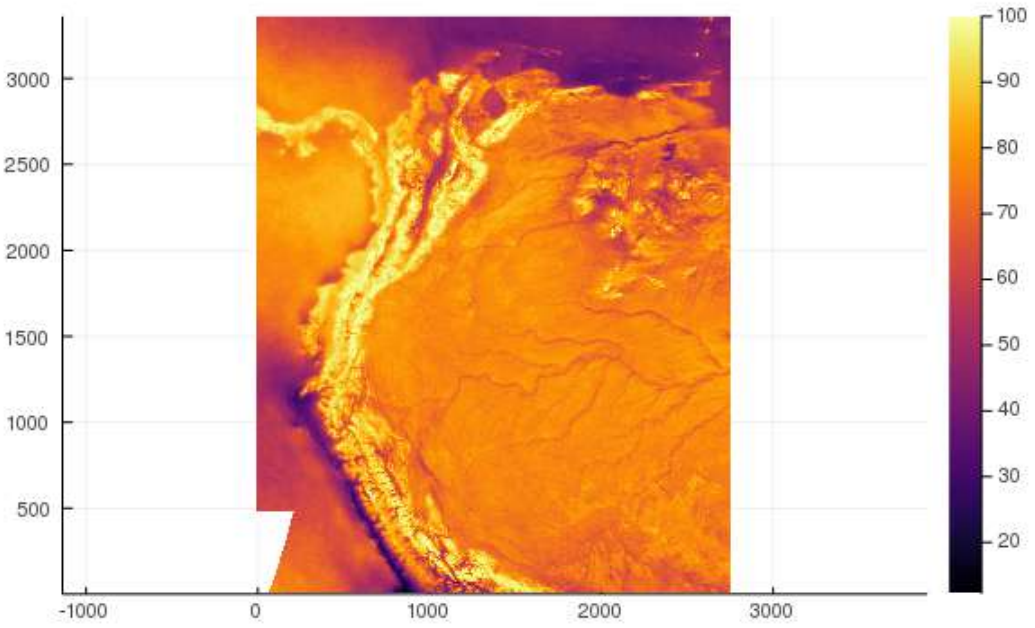
In [29]:
```
ccloudmd = mapCube(medianFx, ccloud, indims=indims, outdims=outdims)
```

```
Progress: 100%|████████████████████████████████████████| Time: 0:01:16
```

Out[29]:
```
In-Memory data cube with the following dimensions
Median                Axis with 1 elements: percentage
Lon                   Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                   Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 44.22 MB
```

In [30]:
```
# Plot data
heatmap(ccloudmd[1,:,:]'[end:-1:1,:], aspect_ratio = :equal)
```

Out[30]:




## Load seasonality ratio

In [31]:
```
mask1
```

Out[31]:
```
Transformed cube ZArray Cube with the following dimensions
Lon                   Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                   Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 44.22 MB
```

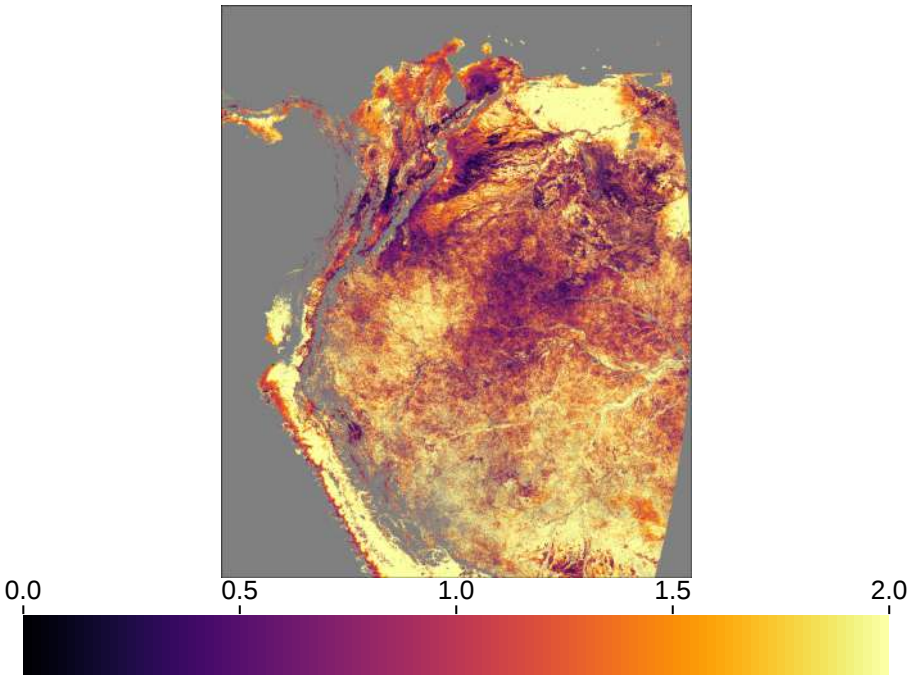
In [32]:
```
ratin = readdlm(string(pathin2, "/monobimodal/dataout/ratio_annual_semiannua
l_maskosc234.csv"),',')
ratmis = map(x->isnan(x) ? missing : x, ratin' |> Array)
rat1 = CubeMem(CubeAxis[getAxis("Lon", c),getAxis("Lat", c)], ratmis)
```

Out[32]:
```
In-Memory data cube with the following dimensions
Lon                   Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                   Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 79.6 MB
```

In [33]:
```
# rat1 = loadCube("ratio_annual_semiannual_pca1kmstd2014")
```

In [34]:
```
plotMAP(rat1, dmin=0, dmax=2)
```

Out[34]:




## Find pixels where variables and the seasonlity ratio map match spatially

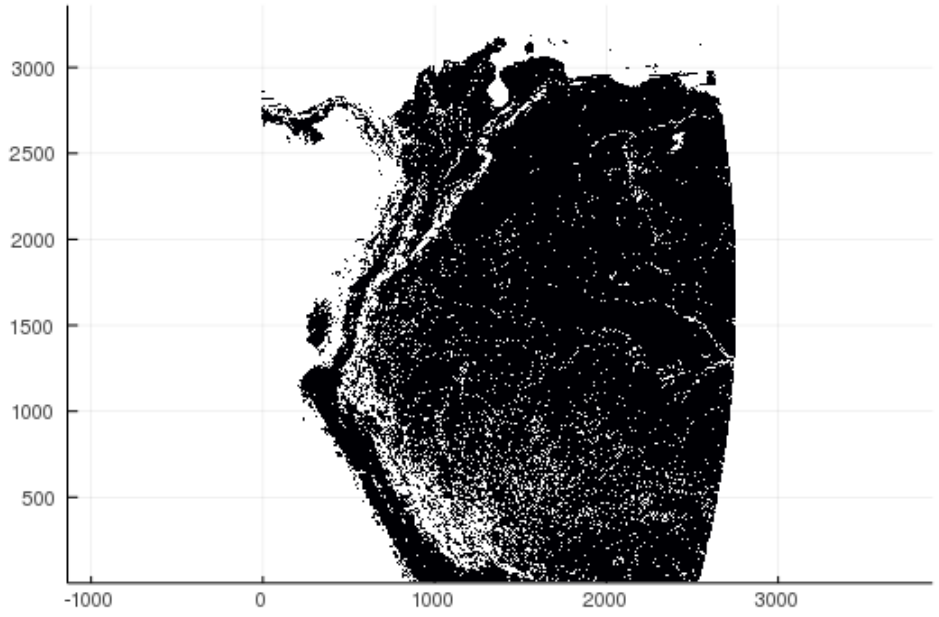
In [35]:
```
# Define a mask for vegetation variables, ratio map, and climate variable
function maskvarratioFx(mask1, rat1, cin)
    maskall = map((x,y,z)->x*y*z, mask1, rat1, cin)
    mask2 = map(x->ismissing(x) ? missing : 1, maskall[:,:]);
end
```

Out[35]:
```
maskvarratioFx (generic function with 1 method)
```

In [36]:
```
# Apply mask to ratio map based on the assessed variable
mask2ppt = maskvarratioFx(mask1, rat1, cppt)
mask2tem = maskvarratioFx(mask1, rat1, ctem)
mask2cloud = maskvarratioFx(mask1, rat1, ccloudmd[Median="percentage"]);
```

In [37]:
```
heatmap(mask2ppt[:,:]'[end:-1:1,:], aspect_ratio = :equal)
```

Out[37]:




```
GKS: Possible loss of precision in routine SET_WINDOW
GKS: Rectangle definition is invalid in routine SET_WINDOW
GKS: Rectangle definition is invalid in routine CELLARRAY
invalid range
```

In [38]:
```
indims = InDims("Lon","Lat")
outdims = OutDims("Lon","Lat")
```

Out[38]:
```
OutDims((ESDL.Cubes.Axes.ByName("Lon"), ESDL.Cubes.Axes.ByName("Lat")), (),
zero, identity, :auto, false, AsArray(), :input, Zarr.NoCompressor(), "", fa
lse, 1)
```

In [39]:
```
function maskout(xout, xin, mask)
    xout[:] = xin.*mask
end
```

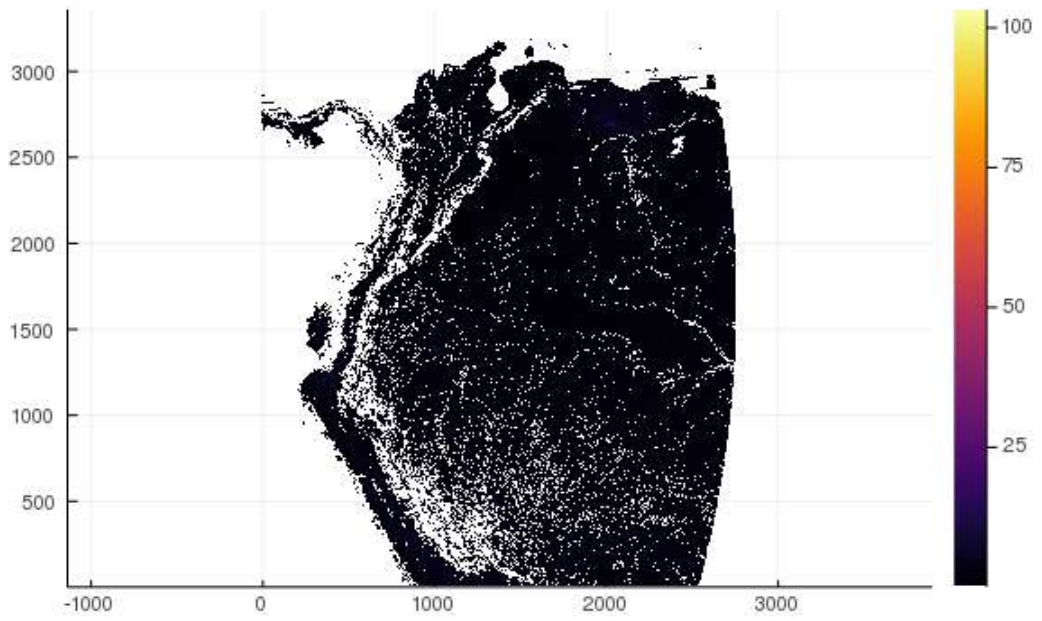
Out[39]:
```
maskout (generic function with 1 method)
```

In [40]:
```
# Exclude pixels from the ratio map that do not match the assessed variable
 using the respective mask
# ratm = mapCube(maskout, rat1, mask2, indims=indims, outdims=outdims)

ratmppt = mapCube(maskout, rat1, mask2ppt, indims=indims, outdims=outdims)
ratmtem = mapCube(maskout, rat1, mask2tem, indims=indims, outdims=outdims)
ratmclouds = mapCube(maskout, rat1, mask2cloud, indims=indims, outdims=outdi
ms)
```

Out[40]:
```
In-Memory data cube with the following dimensions
Lon                  Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                  Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 79.6 MB
```

In [41]:
```
heatmap(ratmppt[:,:]'[end:-1:1,:], aspect_ratio = :equal)
```

Out[41]:

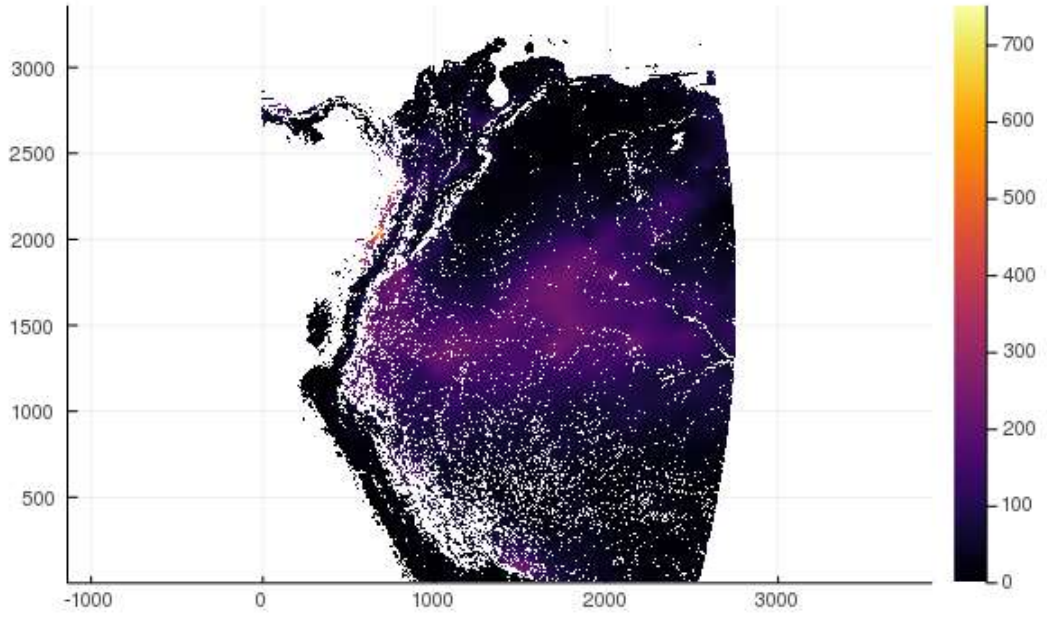



In [42]:
```
# Exclude pixels from the assessed variable that do not match the ratio map
# varm = mapCube(maskout, cvar, mask2, indims=indims, outdims=outdims)

varmppt = mapCube(maskout, cppt, mask2ppt, indims=indims, outdims=outdims)
varmtem = mapCube(maskout, ctem, mask2tem, indims=indims, outdims=outdims)
varmcloud = mapCube(maskout, ccloudmd, mask2cloud, indims=indims, outdims=ou
tdims)
```

Out[42]:
```
In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Median              Axis with 1 elements: percentage
Total size: 44.22 MB
```

In [43]:
```
heatmap(varmppt[:,:]'[end:-1:1,:], aspect_ratio = :equal)
```

Out[43]:




## Select data by biotic unit

In [44]:
```
cbu
```

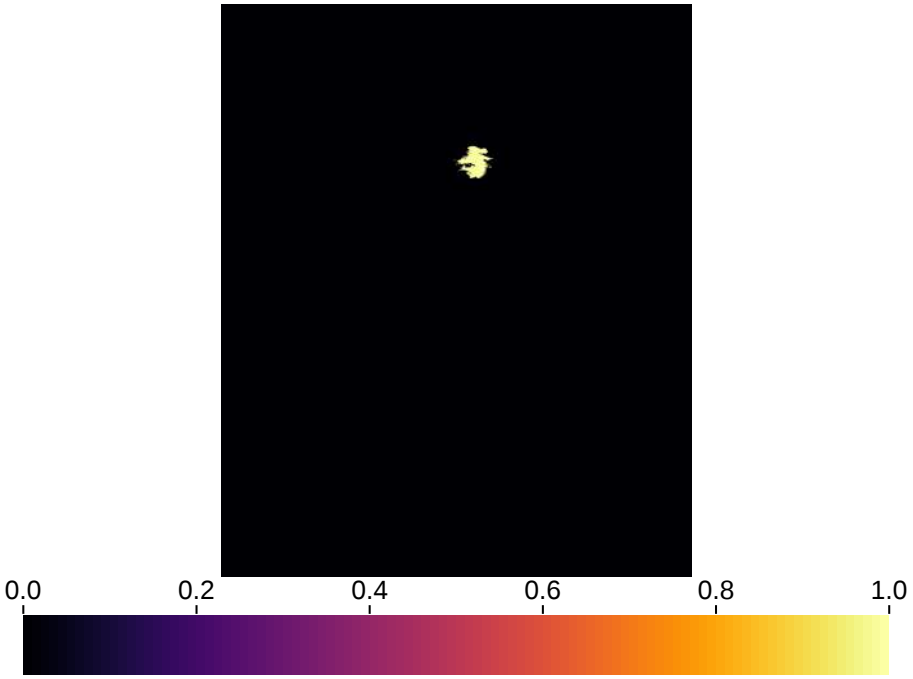
Out[44]:
```
In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
labels: Dict(30.0 => "Bita",47.0 => "Huila-Caqueta",54.0 => "Serrania del Na
quen",32.0 => "Altoandino influencia llanera",50.0 => "Picachos",2.0 => "Est
ribacion norte Sierra Nevada de Santa Marta",40.0 => "Cordillera central",1
1.0 => "Sinu",39.0 => "Casanare",46.0 => "Villavicencio"…)
Total size: 44.22 MB
```

In [45]:
```
i = 27
m1 = map(x -> x==i ? 1.0 : 0.0, cbu)
```

Out[45]:
```
Transformed cube In-Memory data cube with the following dimensions
Lon                 Axis with 2760 Elements from -82.99622135 to -60.0046466
5
Lat                 Axis with 3360 Elements from 13.99613735 to -13.99541735
0000002
Total size: 44.22 MB
```

In [46]:
```
plotMAP(m1)
```

Out[46]:




## Plot one heatmap

In [47]:
```
idx = findall(x-> x==i, cbu[:,:])
cbuin = cbu[:,:]
validx = (map(x->cbuin[x], idx));
```

In [48]:
```
rat1d = ratmppt[:,:];
```

In [49]:
```
varin = varmppt[:,:];
varname = "Precipitation driest month (mm)"
```

Out[49]:
```
"Precipitation driest month (mm)"
```

In [50]:
```
# Number of pixels for selected biotic unit
size(collect(skipmissing(map(x->rat1d[x], idx))))
```

Out[50]:
```
(20560,)
```

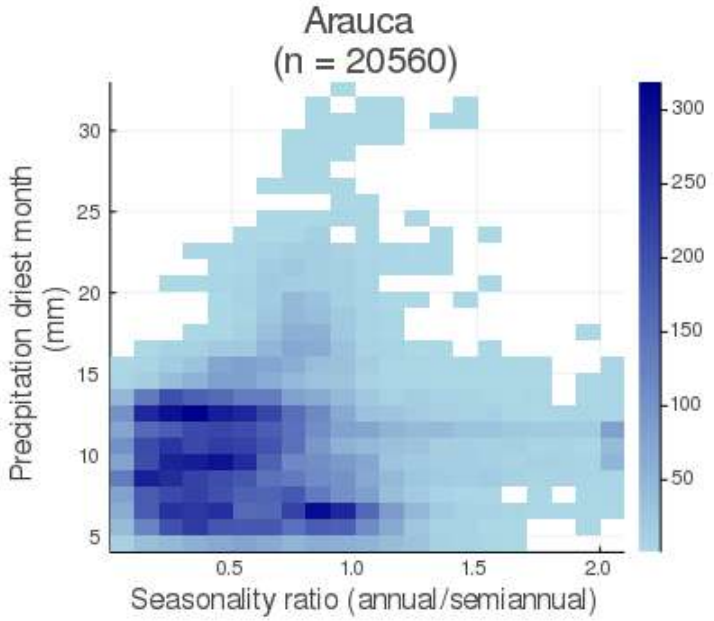
In [51]:
```
ratvaridx = hcat((collect(skipmissing(map(x->rat1d[x], idx)))), (collect(ski
pmissing(map(x->varin[x], idx)))))
#collect(skipmissing(ratidx))
ratvaridx[:,1] = map(x-> x>2 ? 2 : x, ratvaridx[:,1]);
```

In [52]:
```
titleout = size(ratvaridx)[1]
```

Out[52]:
```
20560
```

In [69]:
```
histogram2d(ratvaridx[:,1], ratvaridx[:,2], title = string(bunames[i], " \n
(n = ", size(ratvaridx)[1] , ")"),
    yaxis=(varname), xaxis=("Seasonality ratio (annual/semiannual)"), c=:blu
es)
```

Out[69]:




## Loop for plotting heatmaps with land cover as title

In [82]:
```
function plothmbu(bunames, cbu, rat1d, varin, varname, pathout, yst, yend)

    # From 1 until the length of biotic units names
    for i = 1:66

        titlen = bunames[i]

        # Find the two most dominant land cover clases for the selected biot
ic unit
        lcall = findall(row->row==i, lcbybu2[:,1])
        lcsub = hcat(lcbybu2[lcall,2:end]' |> Matrix, lcbybu[1,2:end])
        lcmis = sortslices(lcsub, dims=1, rev=true)
        idmis = findmax(findall(x->ismissing(x), lcmis[:,1]))[1]
        lcmax = lcmis[idmis+1:end,:]

        # Set figure title based on dominant land covers
        titlex = string(dictlc2[lcmax[1,2]], " (", round(lcmax[1,1], digits=
2), "%) +
",dictlc2[lcmax[2,2]], " (", round(lcmax[2,1], digits=2), "%)")

        # Select pixels of interest bases on the selected biotic unit
        idx = findall(x-> x==i, cbu[:,:])
        ratvaridx = hcat((collect(skipmissing(map(x->rat1d[x], idx)))),
            (collect(skipmissing(map(x->varin[x], idx)))))

        if minimum(ratvaridx[:,2]) < 0
            @show i
        end

    # Plot commands
        pout = histogram2d(ratvaridx[:,1], ratvaridx[:,2],
            titlefontsize = 13, title = string(titlex, "  (n=", size(ratvari
dx)[1] , ")"),
            xlim =(0, 3), xaxis=("Seasonality ratio (annual/semiannual)"),
xtickfont=font(11),
            yaxis=(varname), ytickfont=font(12, "TimesNewRoman"),
#               ylim = (yst, yend), #activate for equal axis range e.g. tempe
rature and clouds
            xguidefont=font(15, "TimesNewRoman"), yguidefont=15,
            dpi=200)

        # Save figure
        nameout2 = join(map(x -> isspace(titlen[x]) ? "" : titlen[x], 1:leng
th(titlen)))
        savefig(pout, string(pathout, "heatmap_bu_", i, "_", nameout2, ".pn
g"))

    end

end
```

Out[82]: plothmbu (generic function with 1 method)

## Figure 8 - Precipitation of the driest month

In [86]:
```
# Variable name for the y-axis label
varname = "Precip. driest month (mm)"

# Location for saving figures
locfol = "minpremonth/axisdiffer/"
pathout = string(pathin2, "dimred/plots/heatmaps/mask/2014/", locfol);
```

In [87]:
```
varname
```

Out[87]:
```
"Precip. driest month (mm)"
```

In [88]:
```
plothmbu(bunames, cbu, ratmppt[:,:], varmppt[:,:], varname, pathout, 0, 800)
```

## Figure S4 - Maximum temperature of the warmest month

In [77]:
```
# Variable name for the y-axis label
varname = "Max. °T warmest month (°C)"

# Location for saving figures
locfol = "maxtemwarmestmon/axisequal/original/"
pathout = string(pathin2, "dimred/plots/heatmaps/mask/2014/", locfol);
```

In [78]:
```
plothmbu(bunames, cbu, ratmtem[:,:], varmtem[:,:], varname, pathout, 0, 35)
```

## Figure S5 - Median annual cloud frecuency

In [79]:
```
# Variable name for the y-axis label
varname = "Median cloud frecuency (%)"

# Location for saving figures
locfol = "clouddays/median/axisequal/original/"
pathout = string(pathin2, "dimred/plots/heatmaps/mask/2014/", locfol);
```

In [80]:
```
plothmbu(bunames, cbu, ratmclouds[:,:], varmcloud[:,:,1], varname, pathout,
40, 100)
```

In [ ]:
```

```