ICIMOD

MANUAL

Selection and downscaling of general circulation model datasets and extreme climate indices analysis

Published by
International Centre for Integrated Mountain Development
GPO Box 3226, Kathmandu, Nepal

ISBN 978 92 9115 6887 (online)

Production team
Shanuj VC (Consultant editor)
Samuel Thomas (Senior editor)
Rachana Chettri (Editor)
Punam Pradhan (Graphic Designer)

Photos
Alex Treadway: p vi
Jitendra Bhajracharya: Cover, pp v, ii, iii, iv
Kham Magar: p vii

Citation
Pradhananga, S., Lutz, A., Shrestha, A., Kadel, I., Nepal, B., & Nepal, S. (2020). *Selection and downscaling of general circulation model datasets and extreme climate indices analysis.* Manual, ICIMOD.

MANUAL

Selection and downscaling of general circulation model datasets and extreme climate indices analysis

Authors

Saurav Pradhananga, Arthur Lutz, Archana Shrestha, Indira Kadel, Bikash Nepal, Santosh Nepal

International Centre for Integrated Mountain Development

Contents

Abbreviations and acronyms

CF	Correction factor
CDD	Consecutive dry days
CDF	Cumulative distribution function
CDO	Climate data operators
CMIP5	Coupled Model Intercomparison Project Phase 5
CSDI	Cold spell duration index
DHM	Department of Hydrology and Meteorology
ECDF	Empirical cumulative distribution function
ETCCDI	Expert team on Climate Change Detection and Indices
FA	Frequency adaptation
GCM	General circulation/climate model
HI-AWARE	Himalayan Adaptation, Water and Resilience
HKH	Hindu Kush Himalaya
ICIMOD	International Centre for Integrated Mountain Development
IDE	Integrated development environment
KNMI	Royal Netherlands Meteorological Institute
MoFE	Ministry of Forests and Environment
NAP	National Adaptation Plan
NetCDF	Network Common Data Form
QM	Quantile mapping
R95pTOT	Precipitation due to very wet days
RCM	Regional Climate Model
RCP	Representative concentration pathway
UTM	Universal Transverse Mercator
WGS	World Geodetic System
WSDI	Warm spell duration index

Acknowledgements

This manual has been prepared as a supplement to the Climate Change Scenarios for Nepal report published by the Ministry of Forests and Environment (MoFE) for the National Adaptation Plan (NAP) Process.

We would like to thank Arun Bhakta Shrestha, Programme Manager, River Basins and Cryosphere, International Centre for Integrated Mountain Development (ICIMOD), and Sanjeev Bhuchar, Theme Leader, Water and Air, ICIMOD for their guidance, comments, and suggestions. We would also like to thank René Wijngaard, Future Water, for helping us prepare and customize the codes for downscaling. We are grateful to Santosh Kaini, Victoria University, Australia; Kabiraj Khatiwada, ICIMOD; and Prerana Amatya, ICIMOD for providing feedback and testing the codes for various regions of the HKH. We thank the Department of Hydrology and Meteorology, Nepal for collaborating with us and supporting us in the preparation of this manual.

Further, we take this opportunity to express our gratitude to our immediate colleagues associated with ICIMOD's Water and Air Theme and River Basins and Cryosphere Regional Programme for their support and cooperation in the completion of this manual.

About climate change scenarios for Nepal

Understanding climatic change, both in terms of its historical and future patterns, is very important for the development of adaptation strategies. The evidences of climate change have been prevalent in Nepal in different sectors. Many studies have suggested that shrinking glaciers, expanding glacial lakes, widespread increase in temperature, and erratic rainfall patterns are some examples of climate change evidences in Nepal. These changes have impacted different sectors such as water resources, biodiversity, ecosystem, agriculture, health and livelihood. Therefore, it is imperative to understand the patterns of climate change so that adaptable solutions can be designed.

To understand the nature of climate change in the future, there are different methods available in literature. They depend on how the general circulation/climate models (GCMs) are selected and downscaled to a finer resolution so that an impact assessment can be carried out. In addition to the downscaling, future climatic extremes were also calculated to aid the adaptation options in the different sectors. The methodological aspects of the process are described in detail by MoFE (2019).

About the manual

This manual has been prepared to select the relevant GCMs by using an envelope-based approach and to carry out an analysis of future changes in precipitation, temperature, and climate extremes. It is the basis for the report called “Climate Change Scenarios of Nepal”, which was published by the MoFE as a part of the NAP process (MOFE, 2019).

The manual provides detailed information about the processes through which the assessment highlighted in the report can be carried out. They include:

- Selection of the GCMs
- Downscaling of the GCM dataset
- Assessment of changes in precipitation and temperature
- Assessment of change in climate extremes

This manual has been used to downscale climate datasets for the Koshi River basin, the Kabul River basin, and the Kailash Sacred Landscape to analyse future scenarios in these basins and the landscape.

The NAP report (MoFE, 2019) can be accessed by clicking on the following links:

- HimalDoc link: http://lib.icimod.org/record/34554
- MoFE link: http://www.mofe.gov.np/downloadfile/MOFE_2019_Climate change scenarios for Nepal_NAP_1562647620.pdf

Targeted users

This manual would be useful for researchers, undergraduates, and graduates who may have to use climate projection datasets for various purposes. With the help of this manual, relevant GCMs for selected catchments/boundaries can be chosen and downscaled to respective areas of interest. It also provides scripts for climate change-related assessment, including for future climatic extremes. By applying these processes and scripts, high-resolution climate scenario datasets can be generated at regional or catchment scales. While this manual has been prepared in the context of Nepal, the process described here is replicable in other parts of the Hindu Kush Himalayan (HKH) region.

Requirements for using the codes in this manual

This section outlines the requirements in terms the software used in this manual.

Statistical computations, including GCM dataset downscaling and analysis of downscaled datasets, are performed using an open-source programming language and software environment, R. A short description about the R language and RStudio is provided below.

R is a system for statistical computation and graphics. It consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files. R was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand (Ikaha and Gentleman 1996). R can be downloaded from the following site: https://cran.r-project.org/src/base/R-3/.

For the Windows-based R, follow this link: https://cran.r-project.org/bin/windows/base/

For more help on the R language, click on this link: https://cran.r-project.org/doc/FAQ/R-FAQ.html#What-is-R_003f

RStudio is a free and open-source integrated development environment (IDE) for R, a programming language for statistical computing and graphics. RStudio can be downloaded from the following links:
https://www.rstudio.com/products/RStudio/#Desktop

https://github.com/rstudio/rstudio

Several specific packages need to be installed for downscaling and analysis. This can be done using this command: install.packages ("insert required package") in the R console. The required packages should be compatible with the installed version of R.

For climate data analysis, Climate Data Operators (CDO), developed by the Max Planck Institute for Meteorology, is used. CDO is a collection of many operators for standard processing of climate and forecast model data. The operators include simple statistical and arithmetic functions, data selection and subsampling tools, and spatial interpolation. CDO is developed for both Linux and Windows systems. There are more than 700 operators that can be calculated using CDO. The Windows version of CDO can be downloaded from the following link: https://code.mpimet.mpg.de/projects/cdo/files

To install CDO, please follow: https://code.mpimet.mpg.de/projects/cdo/wiki/Win32.

The CDO Manual can be downloaded from: https://code.mpimet.mpg.de/projects/cdo/embedded/cdo.pdf

Methodological approach

An advanced envelope-based selection approach described by Lutz et al. (2016) has been used to select and downscale the representative ensemble of GCMs. Here, we provide a short description of the method. Readers are requested to refer to the NAP report for a full description (MoFE, 2019).

GCMs are used to simulate global atmospheric processes. These models are operated at a spatial resolution ranging from approximately 100–250 km^2. Since the GCM datasets are not able to capture regional heterogeneity (such as the atmospheric and orographic processes), these resolutions are too general to carry out any specific assessment at regional scales (such as at the catchment level). Therefore, these GCMs are further downscaled to a finer resolution. The downscaling techniques can be divided into two groups: dynamic downscaling and empirical-statistical downscaling.

Dynamic downscaling uses Regional Climate Models (RCMs) where the GCM usually provides the boundary conditions for an RCM that has a nested domain within the GCM domain, and it operates at a resolution of 10–50 km^2. There are efforts being made to downscale to a much finer resolution. As for empirical-statistical downscaling, it is based on the statistical relationship between large-scale predictors (climate model datasets) and local-scale observations (Fowler et al., 2007; Maraun et al., 2010; Wilby & Wigley, 1997).

There are pros and cons to both dynamic and empirical-statistical downscaling. Dynamic downscaling incorporates complex topography, land–sea contrasts, surface heterogeneities, and detailed physical processes to generate results at higher resolutions. However, it is computationally intensive and requires a high level of expertise to interpret and implement the results. The downscaled product also requires some form of additional bias correction. Statistical downscaling requires relatively low computational power and is easy to interpret and implement. However, it assumes stationarity in the relationship between the GCMs and the observations which might not be always true (Trzaska & Schnarr, 2014).

Among the different statistical downscaling approaches, the quantile mapping (QM) method (Bo et al., 2007; Déqué, 2007) has been found to be the most reliable in mountainous regions (Themeßl et al., 2011b). It has been applied at the catchment scale in the central Himalayas by Immerzeel, Pellicciotti, & Bierkens (2013). Owing to its robustness and good performance in the mountainous areas, the QM approach has been selected to downscale the GCM datasets for Nepal.

QM is based on the principle of comparing distributions of a climatic variable in a dataset of historical observations and climate model control runs, and subsequently defining an error function to correct for biases for each quantile in the distribution. This error function is applied to a future climate model run to correct the future climate dataset. The approach can be based on empirical or fitted probability distributions (Piani et al. 2010; Themeßl et al. 2011a). In this manual, we have used the empirical probability distribution function to correct the GCM dataset. The four step followed in this approach is shown in figure 1.

FIGURE 1 GCM SELECTION AND DOWNSCALING PROCEDURE

INITIAL POOL OF CLIMATE MODELS

RCP4.5: 105 GCM runs
RCP8.5: 77 GCM runs

STEP 1

Initial selection based on changes in mean air temperature and annual precipitation sum

REMAINING

RCP 4.5: 20 GCM runs
RCP 8.5: 20 GCM runs

STEP 2

Refined selection based on projected changes in four indices of future climate extremes

REMAINING

RCP 4.5: 8 GCM runs
RCP 8.5: 8 GCM runs

STEP 3

Final selection based on model skills in simulating annual cycle of air temperature and precipitation

FINAL MODEL ENSEMBLES

RCP 4.5: 4 GCM runs (~250 km)
RCP 8.5: 4 GCM runs (~250 km)

STEP 4

Downscaling of selected GCMs using quantile mapping for Nepal

DOWNSCALED MODEL ENSEMBLES

RCP 4.5: 4 GCM runs (10 km)
RCP 8.5: 4 GCM runs (10 km)

Source: Adapted from Lutz et. al 2016

Step 1: Data download and initial selection based on changes in the average annual mean air temperature and the average annual total precipitation

As many as 105 GCMs for Representative Concentration Pathway (RCP) 4.5 and 77 GCMs for RCP8.5 have been taken into consideration for the selection of representative GCMs for Nepal. In the first step, we reduce the number of GCMs to 20 (5 model runs × 4 corners = 20 model runs for each RCP) as the representative model for Nepal. The 4 corners are representative of the spectrum of projections for temperature and precipitation change, i.e., "cold, dry", "cold, wet", "warm, dry", and "warm, wet". These corners are determined by calculating the 10th and 90th percentile values of the average annual mean air temperature (ΔT) and the average annual total precipitation (ΔP) of the GCMs under RCP4.5 and RCP8.5 scenarios after resampling all the GCM datasets to the same 2.5°× 2.5° grid.

The 10th percentile value for ΔT and the 10th percentile value for ΔP are in the "cold, dry" corner of the spectrum. The 10th percentile value for ΔT and the 90th percentile value for ΔP are in the "cold, wet" corner of the spectrum. The 90th percentile value for ΔT and the 10th percentile value for ΔP are in the "warm, dry" corner of the spectrum. The 90th percentile value for ΔT and the 90th percentile value for ΔP are in the "warm, wet" corner of the spectrum. The 10th and 90th percentile values are chosen rather than the minimum and maximum projections to avoid selecting outliers (e.g. (Immerzeel et al., 2013)).

The range of projected changes in area averaged annual mean air temperature (ΔT) and the average annual total precipitation (ΔP) between 1981–2010 and 2036–65 for the whole of Nepal is calculated. The proximity of the model runs to the 10th and 90th percentile values is then derived from the model runs' percentile rank scores corresponding to their projections for ΔT and ΔP with respect to the entire range of projections in the entire ensemble:

$$D_{p_j^P, p_j^T} = \sqrt{(P_i^P - P_j^P)^2 + (P_i^T - P_j^T)^2}$$

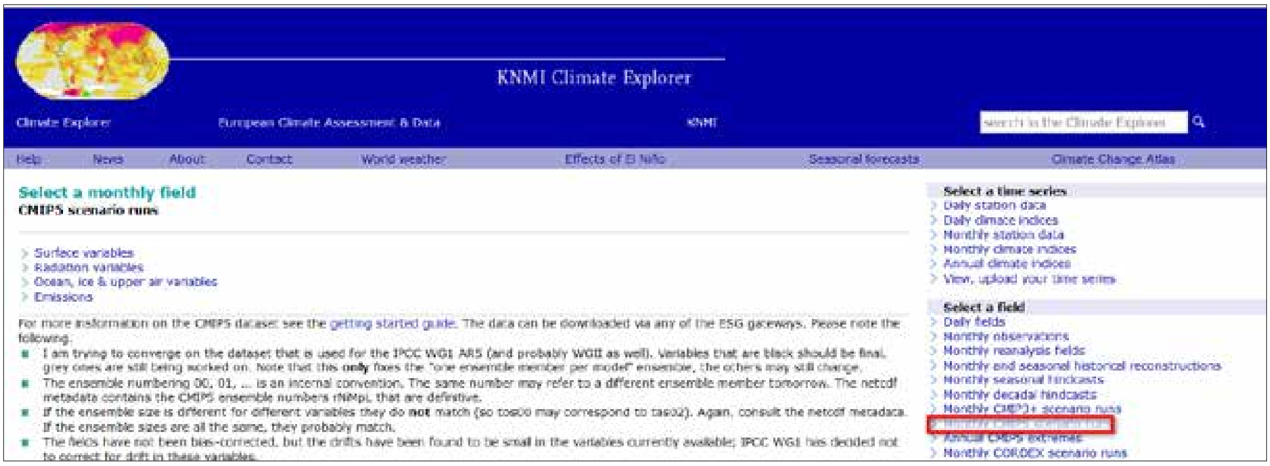
Where $D_{p_j^P p_j^T}$ is the distance of a model (j)'s ΔT and ΔP (P_j^T and P_j^P, respectively) to the corner (i)'s 10th and/or 90th percentile score of ΔT and ΔP for the entire ensemble (P_j^T and P_j^P, respectively). For each corner, 5 models with the lowest values for $D_{p_j^P p_j^T}$ and outputs available at a daily time step are selected from the ensemble. (Note: Models with data available at the daily time step are selected because this is a requirement for the empirical-statistical downscaling method to be applied to the GCM runs in Step 4). Nonetheless, all model runs are included in the initial pool of available model runs used to calculate the runs' percentile scores to have a complete representation of all the projected possible futures.

The first process involved in Step 1 is to download the area averaged monthly mean temperature and monthly total precipitation timeseries datasets and calculate the delta changes for future periods of GCMs with respect to their historical period. The steps to be followed are provided below along with the sub-steps involved (to download datasets, registration is required):

Here we have taken precipitation as an example.

1. First go to the KNMI (Koninklijk Nederlands Meteorologisch Instituut – Royal Netherlands Meteorological Institute) Climate Explorer website (http://climexp.knmi.nl).
2. Choose "Monthly CMIP5 scenario runs" from "Select a field" on the right side (the red box in figure 2).

FIGURE 2 MONTHLY CMIP5 SCENARIO RUNS DATA DOWNLOAD INTERFACE



3. Then select all the members of the GCMs for the required variable (e.g., "pr" for precipitation) for a given RCP as shown in figure 3 (1). Members are defined as all the runs for a given model with different realizations, initializations, and physics. In the example below, for RCP4.5, 105 members of all models have been selected. After the selection of the model, choose "Select field" at the top of the interface (2).

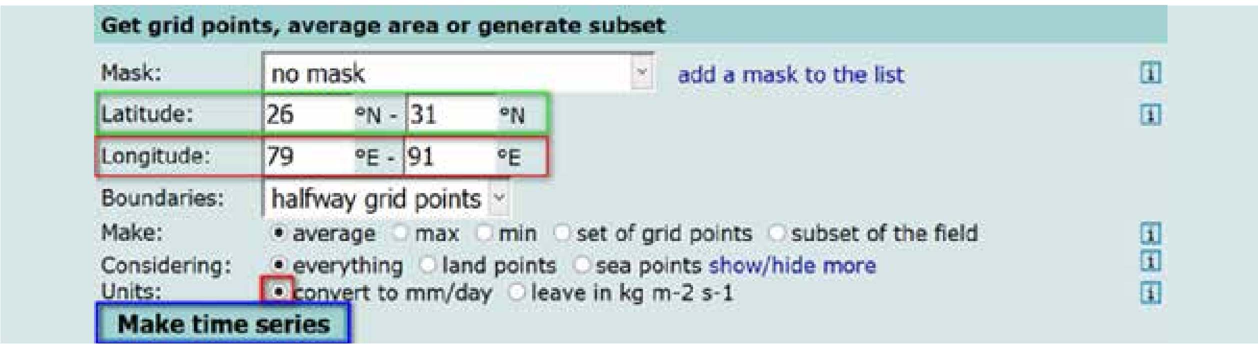
FIGURE 3 VARIABLE AND GCM MEMBERS' SELECTION INTERFACE

Select field Choose a field and press this button 2

model	exp	tas	tas min	tas max	pr	evsp sbl	pme	hurs	taz	psl
all models	rcp26	32	31	31	32	30	30	27	31	32
	rcp45	42	41	41	42	40	40	35	42	42
	rcp60	25	23	23	25	24	24	24	25	25
	rcp85	39	38	38	39	37	37	32	37	39
	rcp45to85	106	102	102	106	101	101	91	104	106
	piControl	43	38	38	39	38	38	32		39
one member per model	rcp26	32	31	31	32	30	30	27	31	32
	rcp45	42	41	41	42	40	40	35	41	42
	rcp60	25	23	23	25	24	24	24	25	25
	rcp85	39	38	38	39	37	37	32	37	39
	rcp45to85	106	102	102	106	101	101	91	103	106
	piControl	41	37	37	38	37	37	31		38
all members	rcp26	65	64	64	65	62	62	33	63	65
1	rcp45	108	103	105	105	101	100	89	102	106
	rcp60	47	43	43	47	46	46	44	47	47
	rcp85	81	75	76	77	73	72	64	72	78
	rcp45to85	236	221	224	229	220	218	197	221	231
	piControl	43	38	38	39	38	38	32		39

4. In the next window, the region of the study area should be entered by giving a range of latitude and longitude. The latitude and longitude of Nepal are provided in figure 4 as an example. Please select "convert to mm/day" in units in case of precipitation, and "convert to °C" in units in case of temperature. Then click on "Make time series". This will generate the area averaged monthly precipitation timeseries for all the GCMs.

FIGURE 4 STUDY AREA SELECTION INTERFACE



5. The calculation of the monthly precipitation of the study area for all the available members in the website is done remotely in the server of KNMI (figure 5).

FIGURE 5 **CALCULATION PROGRESS INTERFACE**

data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 000.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 001.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 002.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 003.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 004.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 005.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 006.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 007.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 008.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 009.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 010.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 011.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 012.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 013.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 014.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 015.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 016.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 017.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 018.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 019.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 020.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 021.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 022.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 023.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 024.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 025.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 026.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 027.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 028.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 029.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 030.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 031.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 032.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 033.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 034.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 035.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 036.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 037.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 038.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 039.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 040.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 041.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 042.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 043.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 044.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 045.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 046.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 047.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 048.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 049.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 050.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 051.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 052.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 053.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 054.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 055.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 056.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 057.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 058.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 059.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 060.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 061.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 062.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 063.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 064.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 065.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 066.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 067.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 068.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 069.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 070.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 071.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 072.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 073.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 074.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 075.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 076.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 077.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 078.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 079.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 080.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 081.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 082.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 083.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 084.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 085.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 086.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 087.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 088.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 089.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 090.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 091.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 092.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 093.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 094.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 095.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 096.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 097.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 098.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 099.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 100.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 101.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 102.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 103.dat, data/icmip5 pr Amon ens rcp45 79-91E 26-31N n su 104.dat, (eps, pdf, raw data)

6. To obtain the area averaged monthly total precipitation timeseries for all members of RCP4.5 from the KNMI website, we use the following R code. Similarly, temperature timeseries can also be downloaded.

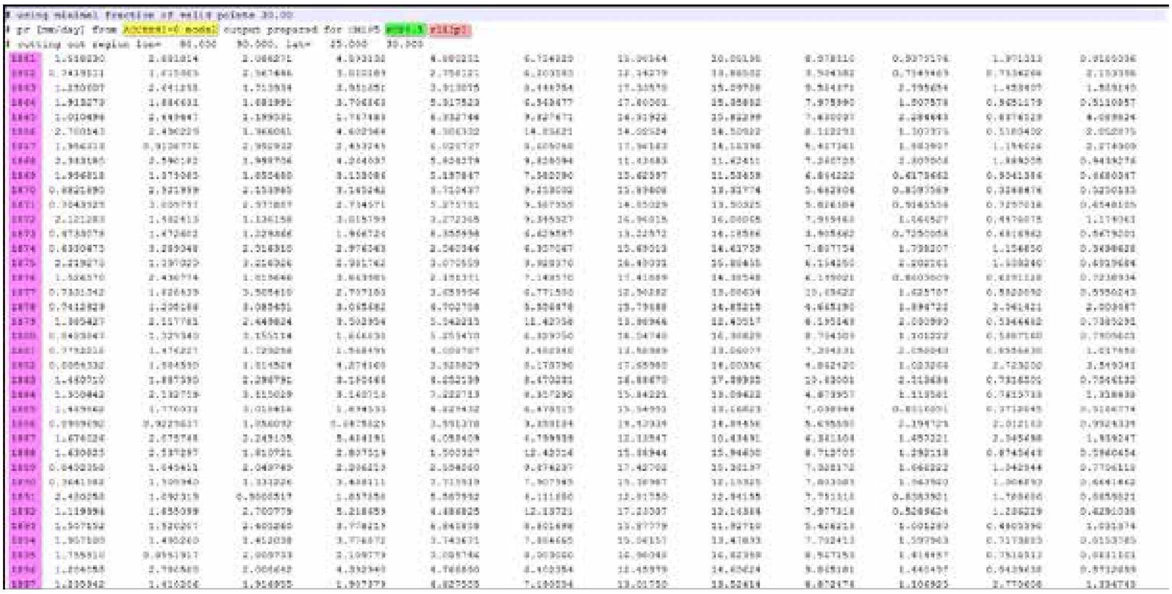
```
--------------------------------------------Code begins--------------------------------------------
## Script to download precipitation timeseries from KNMI website

rm(list = ls()) #removes all stored variable from R enviroment
#provide a folder to store the monthly precipitation for all members from the KNMI’s server
output_folder <- “C:\\Output\\”

#i is the number of members available, 105 in our example for Nepal
#(note that the numbering start from zero)
for (i in seq(0,104))
{
  inum <- as.numeric(i)
  istring <- formatC(inum, width=3, flag=0)
  #change file name in the url below according to the variable and coordinates
  url <- paste(“https://climexp.knmi.nl/data/icmip5_pr_Amon_ens_rcp45_80-90E_25-30N_n_su_”,
  istring,”.dat”,sep=””)
  #generate names of file to be downloaded in the output folder
  output_file <- paste(output_folder,”pr_rcp45_Nepal_ensemble_member”,istring,”.dat”, sep=””)
  #command to download the file
  download.file(url, output_file, method=”auto”, quiet = FALSE, mode = “w”, cacheOK = TRUE)
  #checking progress
  print(istring)
}
print(“finished”)
---------------------------------------------Code ends--------------------------------------------
```

7. The average annual mean air temperature and the average annual total precipitation for all years are calculated from the downloaded monthly datasets. The annual datasets are then averaged over the time period (30 years) from which the changes in ΔT and ΔP between the reference period (1981–2010) and the future period (2036–65) are calculated (in this example). Time periods can be changed as per the user's need. A sample of the downloaded dataset is shown in figure 6.

FIGURE 6 **EXAMPLE DATASET OF A MEMBER OF ONE GCM**



Here, the main consideration is to take note of the model, the RCP and the ensemble. The file consists of year in the first column and the remaining columns show the monthly average data for January to December for the corresponding year. The monthly data can then be averaged (for temperature) or summed (for precipitation) to obtain the total annual value. Then changes in ΔT and ΔP between the reference period (1981–2010) and the future period (2036–65) can be calculated.

Below is the code for calculating the delta changes in precipitation and temperature.

```
---------------------------------------------Code begins---------------------------------------------
## Script to calculate delta changes in precipitation and temperature

rm(list = ls())

##Settings
# provide path to input folder with downloaded files from previous step
input_folder <- "C\\Input\\"
# provide path to output folder
output_folder <- "C:\\Output\\"
# set working directory
setwd(input_folder)


# list RCPs
rcps <- c('rcp45','rcp85')

# list variables
vars <- c('pr','tas')

# days in each month (Jan to Dec)
m_day <- as.matrix(c(31,28,31,30,31,30,31,31,30,31,30,31))

# reference period
ref_startyear <- 1981
ref_endyear <- 2010
# future period
fut_startyear <- 2036
```

```
fut_endyear <- 2065
##Settings end
for (var in vars)
{
# list file with variable as the pattern in the filename
{
# grep command seperates the precipitation files according to the RCP
  infile <- abc[grep(rcp,abc)]

# create an empty dataframe to store RCP, model name, and delta change values for precipitation file
  summary <- data.frame(RCP=character(1),Model=character(1),Delta=numeric(1), stringsAsFactors=FALSE

 for (i in seq(1,length(infile)))
  {
    inum <- as.numeric(i-1)
    istring <- formatC(inum, width=3, flag=0)

   # read the comments line of the data to store model name, RCP and ensemble
    # check for the number of lines to be skipped
    modelline <- scan(infile[i], '', skip = 1, nlines = 34, sep = '\n')
    line <- modelline[grep("operating",modelline)]
    linesplit <- unlist(strsplit(line,split=" "))

    # store the model name
    model <- linesplit[5]
    # store the ensemble
    ensemble <- linesplit[12]

    # read monthly data in to a data frame
    data = read.table(infile[i], sep="",
    col.names=c("year","Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"))

    # remove year column from dataset
    data_1 <- as.matrix(data[,-1])

    # calculate weighted sum to calculate annual total precipitation for each year
    if (var=="pr") {y_data <- data_1%*%m_day}

    # calculate weighted average to calculate average annual mean temperature for each year
    if (var=="tas") {y_data <- data_1%*%m_day/365}

    # combine yearly data to corresponding year
    annual <- cbind(data[1],y_data)

    # calculate mean for each reference period
    prref <- annual[annual$year<=ref_endyear & annual$year>=ref_startyear,]
    prref <- mean(prref[,2])

    # calculate mean for each future period
    prfut_f <- annual[annual$year<=fut_endyear & annual$year>=fut_startyear,]
    prfut_f <- mean(prfut_f[,2])

    # calculatae delta change in future period from reference period
    if (var=="pr") {delta_f <- round(((prfut_f/prref)*100) - 100,digits = 2)}
    if (var=="tas") {delta_f <- round(prfut_f-prref,digits = 2)}

     row <- c(rcp,paste(model,ensemble, sep="_"),delta_f)

    # stack delta change of all models
    summary <- rbind(summary,row)
  }

# removing empty row
  summary <- summary[-1,}

   outfile <- paste(output_folder,rcp,"_",var,"_Nepal_all_members_delta_1981_2010_2036_2065.
csv",sep="")

# writing output to a CSV file
  write.csv(summary,outfile)

print("finished")
}
--------------------------------------------Code ends--------------------------------------------
```

The average annual value and delta changes for mean temperature can also be calculated following the same procedure as that for precipitation.

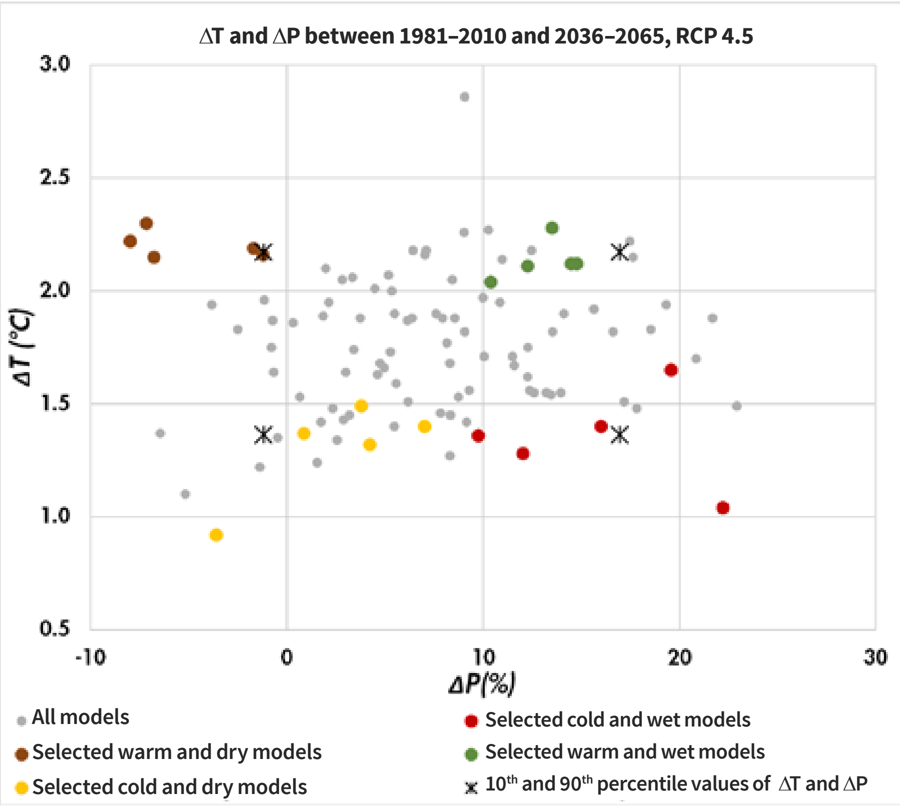
8. A sample of the delta changes for annual precipitation (%) and annual mean temperature (°C) calculated from Step 7 is presented in Table 1:

TABLE 1 SAMPLE ΔP AND ΔT FOR 7 OF THE 105 MODELS

Model	Delta P	Delta T
ACCESS1-0_r1i1p1	5.57	1.59
ACCESS1-3_r1i1p1	9.15	1.42
bcc-csm1-1_r1i1p1	16	1.4
bcc-csm1-1-m_r1i1p1	-0.79	1.75
BNU-ESM_r1i1p1	8.15	1.77
CanESM2_r1i1p1	14.77	2.12
CanESM2_r2i1p1	14.48	2.12

From all the delta values of P and T for all the model runs, the 10th and 90th percentile values are calculated. The respective distance ($D_{p_j^P p_j^T}$) is calculated for each model from 4 corners to find the 5 closest models to each of the corners. These will be the 20 models selected for the next step (4 corners X 5 model runs = 20 GCMs). The models selected for RCP4.5 for Nepal are shown in figure 7.

FIGURE 7 INITIAL MODEL SELECTION BASED ON CHANGES IN THE AVERAGE ANNUAL PRECIPITATION AND THE AVERAGE ANNUAL MEAN TEMPERATURE



Source: MoFE 2019

Step 2: Refined selection based on projected changes in four indices of future climate extremes

In this step, the model runs are evaluated for their projected changes in future climate extremes. The changes in future climate extremes are evaluated by considering the changes in two indices each for air temperature and precipitation (Peterson 2005; see Table 2) of the ETCCDI (Expert Team on Climate Change Detection and Indices) for both air temperature and precipitation. For the characterization of changes in air temperature extremes, the changes in the warm spell duration index (WSDI) and the cold spell duration index (CSDI) are analysed. For the characterization of changes in precipitation extremes, the precipitation due to extremely wet days (R95pTOT) and the number of consecutive dry days (CDD) are considered.

The changes in these indices between the future period (2036– 65) and the reference period (1981–2010) are calculated from the database constructed by Sillmann et al. (2013a, 2013b). As this database does not contain all the GCM runs used for the initial selection, the indices of the ETCCDI for those GCMs are calculated using the same procedures as Sillmann et al. (2013a, 2013b) used in their study. The indices are calculated from the daily model output for each individual year in the future and reference periods, for the individual 2.5°× 2.5° grid cells covering the study area. For both the periods, the indices are then averaged over a period of 30 years. The changes in the indices are later calculated as a percentage change for the future period with respect to the reference period.

Subsequently, these changes in the indices are averaged over the 2.5°× 2.5° grid cells covering the study area. For each model chosen during the initial selection, the most relevant index for air temperature and the most relevant index for precipitation are considered. For example, for the models in the "warm, wet" corner, WSDI – indicating warm spells – and R95pTOT –indicating extreme precipitation events – are considered; whereas CDD and CSDI are considered in the dry and cold corners.

For each corner, the two relevant indices are both ranked and given scores from 1–5. The largest difference scores 5 whereas the smallest scores 1 for that index. Both scores are then averaged to obtain a final score. The models with the two highest combined scores are thus selected for the next step. For each RCP, 4 corners × 2 (at least) models = 8 models (at least) are selected, which are validated to the climatic reference product in the next step.

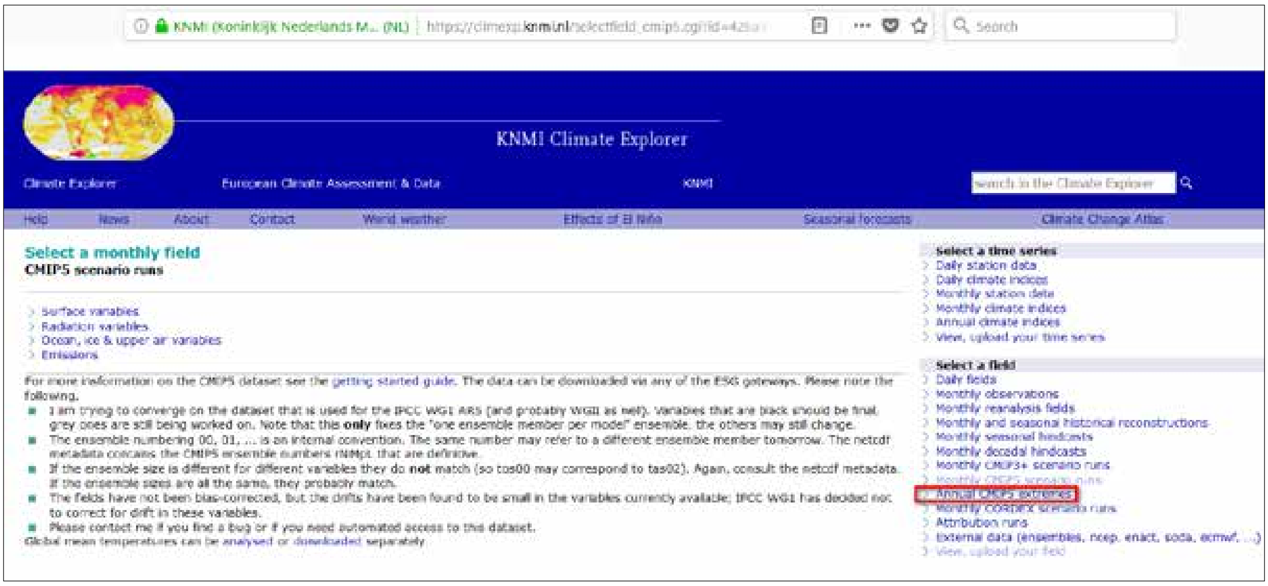
The detailed process involved in Step 2 to download the area averaged extreme indices timeseries datasets of GCMs for historical and projected future periods are provided below with the number of sub-steps:

Here we have taken CDDs as an example.

1. First go to the KNMI Climate Explorer website (http://climexp.knmi.nl).
2. Choose 'Annual CMIP5 extremes' from "Select a field" on the right side (the red box in figure 8).

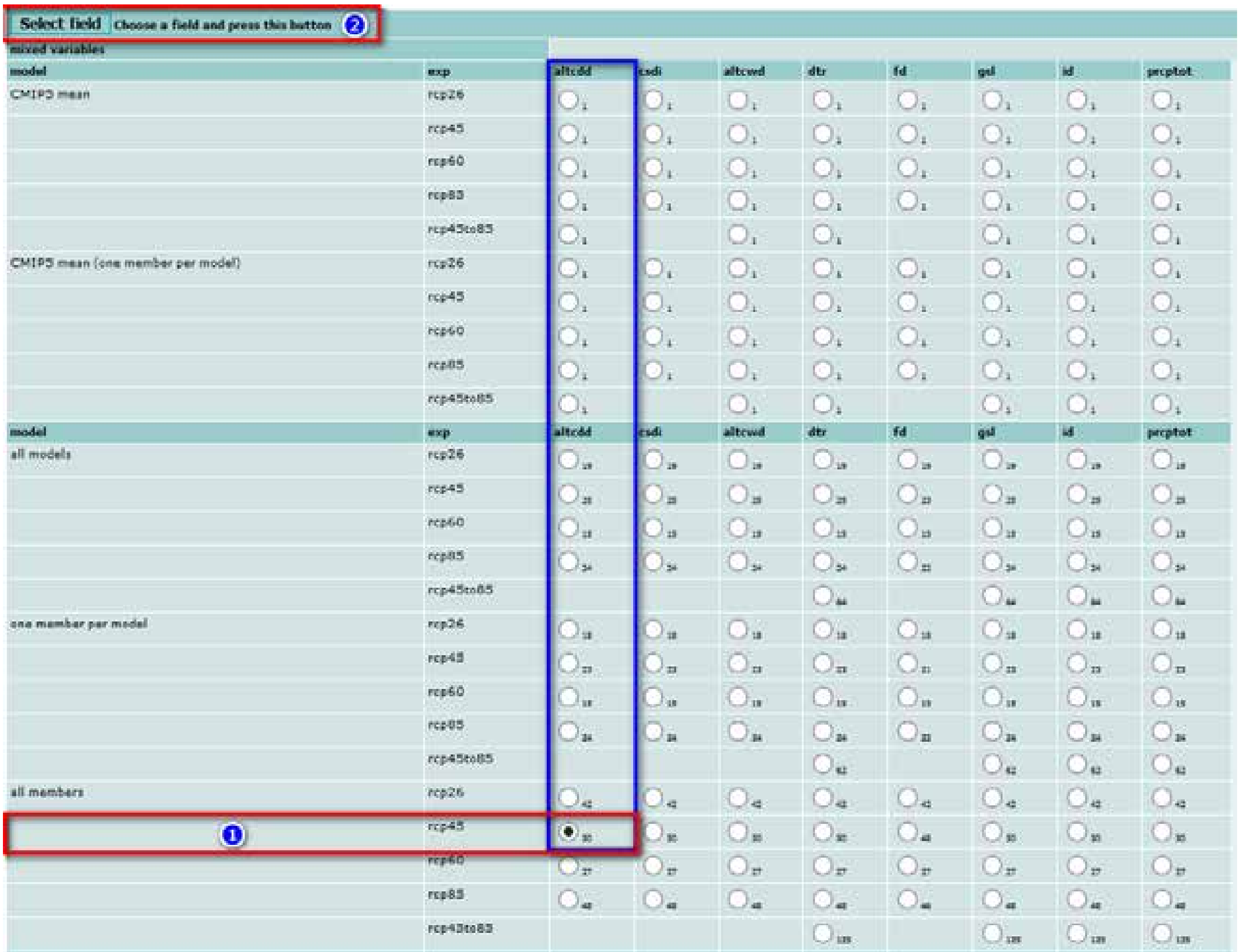
TABLE 2	DESCRIPTION OF INDICES OF ETCCDI USED IN STEP 2	
Meteorological variables	**ETCCDI index**	**Index description**
Precipitation	R95pTOT	Precipitation due to very wet days (> 95th percentile)
Precipitation	CDD	Consecutive dry days: maximum length of a dry spell (P < 1 mm)
Air temperature	WSDI	Warm spell duration index: count of days in a span of at least 6 days where TX_{ij} > 90th percentile (TX_{ij} is the daily maximum temperature on day i in period j)
Air temperature	CSDI	Cold spell duration index: count of days in a span of at least 6 days where TN_{ij} < 10th percentile (TN_{ij} is the daily minimum temperature on day i in period j)

FIGURE 8 ANNUAL CMIP5 EXTREME DATA DOWNLOAD INTERFACE



3. Then select all members of the GCMs for the required indices (e.g., “altcdd” for CDD) for a given RCP as shown in figure 9 (1). In the example below, for RCP4.5, 50 members of all models are selected. After the selection of the model, choose “Select field” from the top of the interface (2).

FIGURE 9 VARIABLE AND GCM MEMBERS’ SELECTION INTERFACE FOR EXTREME INDICES

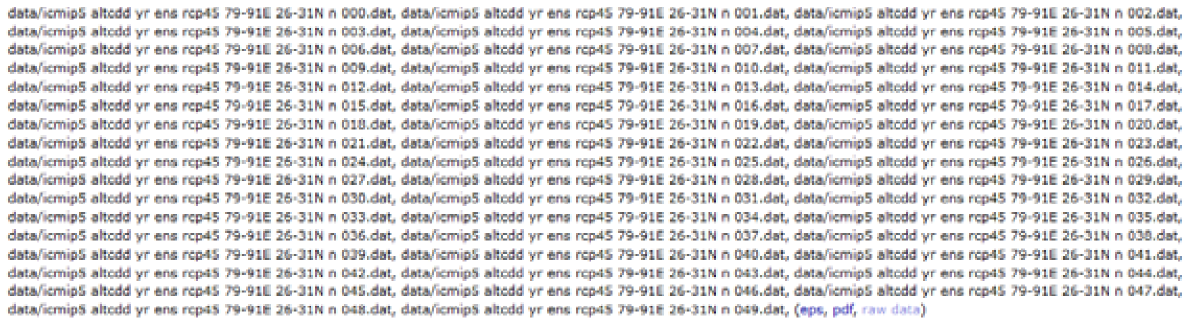


4. In the next window, the region of the study area should be entered by giving a range of latitude and longitude. The latitude and longitude of Nepal are provided in figure 10 as an example. Click on "Make time series". This will generate the area averaged CDD timeseries for all GCMs.

FIGURE 10 **STUDY AREA SELECTION INTERFACE FOR EXTREME INDICES**

5. The calculation of CDDs of the study area for all the available members of the website is done remotely in the server of KNMI (figure 11).

FIGURE 11 **CALCULATION PROGRESS INTERFACE FOR EXTREME INDICES**



data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 000.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 001.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 002.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 003.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 004.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 005.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 006.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 007.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 008.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 009.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 010.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 011.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 012.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 013.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 014.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 015.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 016.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 017.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 018.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 019.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 020.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 021.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 022.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 023.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 024.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 025.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 026.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 027.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 028.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 029.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 030.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 031.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 032.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 033.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 034.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 035.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 036.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 037.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 038.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 039.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 040.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 041.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 042.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 043.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 044.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 045.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 046.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 047.dat,
data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 048.dat, data/icmip5 altcdd yr ens rcp45 79-91E 26-31N n 049.dat, (eps, pdf, raw data)

6. To obtain the area averaged CDD timeseries for all members of RCP4.5 from the KNMI website, we use the following R code.

```
--------------------------------------------Code begins--------------------------------------------
rm(list = ls()) #removes all stored variable from R environment
#provide a folder to store the CDD values for all members from the KNMI's server
output_folder <- "C:\\Output\\"
#n is the number of members available minus 1, No. of model for Nepal is 50
#(note that the numbering start from zero)

n <- 49
for (i in seq(0,n))
{
  inum <- as.numeric(i)
  istring <- formatC(inum, width=3, flag=0)

  #change file name in the url below according to variable and coordinates
  url <- paste("https://climexp.knmi.nl/data/icmip5_altcdd_yr_ens_rcp45_79-91E_26-31N_n_",
  istring,".dat",sep="")

  #generate names of file to be downloaded in the output folder
  output_file <- paste(output_folder,"CDD_rcp45_Nepal_ensemble_member",istring,".dat", sep="")

  #command to download the file
  download.file(url, output_file, method="auto", quiet = FALSE, mode = "w", cacheOK = TRUE)

  #checking progress
  print(istring)
}
print("finished")
--------------------------------------------Code ends--------------------------------------------
```

7. The annual datasets are then averaged over the time period from which the changes in the extreme indices between the reference period (1981–2010) and the future period (2036–65) are calculated. The time periods for delta calculation can be changed as per the user's need. A sample of the downloaded dataset is shown in figure 12.

FIGURE 12 EXAMPLE DATASET OF CDD FOR A MEMBER OF ONE GCM

```
# using minimal fraction of valid points 30.00
# altcddETCCDI [days] from ETCCDI indices computed on ACCESS1-0 model output prepared for CMIP5 RCP4.5 r1i1p1
# cutting out region lon=   80.000   90.000, lat=   25.000   30.000
1861   27.00133
1862   31.44474
1863   21.64795
1864   32.91586
1865   30.70757
1866   27.29692
1867   29.96630
1868   19.31323
1869   35.44172
1870   39.29951
1871   28.80441
1872   31.77163
1873   42.52572
1874   32.59497
1875   24.35867
1876   34.79123
1877   36.35400
1878   31.82838
1879   26.90469
1880   36.89227
1881   41.61966
1882   30.56350
1883   23.28246
1884   25.48520
1885   39.48950
1886   44.14978
1887   21.41984
```

Here, the main consideration is to take note of the model, the RCP and the ensemble. The file consists of year in the first column, and the second column shows the annual data for the corresponding year. Since the data is available at the annual time scale, delta change in the extreme indices can be calculated directly using following code. The average annual value and delta changes for R95pTOT, WSDI and CSDI can also be calculated following the same procedure as that for CDD.

```
-------------------------------------------Code begins-------------------------------------------
rm(list = ls())
# provide path to input folder with downloaded files from previous step
input_folder <- "C\\Input\\"
# provide path to output folder
output_folder <- "C:\\Output\\"
# set working directory
setwd(input_folder)

# list RCPs
rcps <- c('rcp45','rcp85')
# climatic extreme indices used in the calculation
clim_ext <- c('CDD','r95p','CSDI','WSDI')

for (rcp in rcps)
 {
  # list files according to respective rcp in the filename
  if(rcp == "rcp45") {rcp_file <- list.files(path=input_folder,pattern = "rcp45",full.names = T)}
  if(rcp == "rcp85") {rcp_file <- list.files(path=input_folder,pattern = 'rcp85',full.names = T)}
  for (c_e in clim_ext)
```

```
{
    # grep command seperates the files according to the indices
    infile <- rcp_file[grep(c_e,rcp_file)]
    summary <- data.frame(RCP=character(1),Model=character(1),Delta=numeric(1), stringsAsFactors=-
FALSE)
    for (i in seq(1,length(infile)))
    {
      inum <- as.numeric(i-1)
      istring <- formatC(inum, width=3, flag=0)

      # read the second line of the data to store model name, RCP and ensemble
      # check for the number of lines to be skipped
      line <- scan(infile[i], '', skip = 1, nlines = 1, sep = '\n')
      linesplit <- unlist(strsplit(line,split=" "))

      # store the model name
      model <- linesplit[9]

      # store the ensemble
      ensemble <- linesplit[16]

      # read area averaged annual data in to a data frame
      data = read.table(infile[i], sep="", col.names=c("year","value"))

      # calculate mean for each reference period
      prref <- data[data$year<2006 & data$year>1980,]
      prref <- mean(prref[,2])

      # calculate mean for each future period
      prfut_f <- data[data$year>2035 & data$year<2066,]
      prfut_f <- mean(prfut_f[,2])

      # calculatae delta change in future period from reference period
      delta_f <- round(((prfut_f/prref)*100) - 100,digits = 2)

      row <- c(rcp,paste(model,ensemble, sep="_"),delta_f)

      # stack delta change of all models row by row
      summary <- rbind(summary,row)
    }

    # removing empty first row
    summary <- summary[-1,]

    #generate output filename
    outfile <- paste(output_folder,c_e,"_",rcp,"_Nepal_all_members_delta_1981_2010_2036_2065.
csv",sep="")

    # writing output to a CSV file
    write.csv(summary,outfile)

    #check progress
    print("Working on it")
  }
}

print("finished")
-----------------------------------------------Code ends---------------------------------------------
```

After calculating the delta changes for relevant indices for each corner, a final score is obtained by averaging those rank. Based on the final score, the two models with the highest scores are selected (Table 3). Here, the models selected for Step 3 are highlighted in a shade of green.

TABLE 3 **GCM RUNS ANALYSED DURING THE REFINED SELECTION STEP 2**

RCP	Projection	Model	ΔP (%)	ΔT (°C)	ΔCSDI (%)	ΔCDD (%)	ΔWSDI (%)	ΔR95p (%)	P_{index} Rank	T_{index} Rank	Combined Score
RCP4.5	Cold, Dry	NOAA_GFDL_GFDL-ESM2M_r1i1p1	0.87	1.37	-52.4	4.3	308.2	19.4	3	3	3
		inmcm4_r1i1p1	-3.59	0.92	-24.2	11.4	182.1	1.9	1	5	3
		NOAA_GFDL_GFDL-ESM2G_r1i1p1	4.23	1.32	-49.0	-2.0	372.6	21.5	2	1	1.5
		CCSM4_r1i1p1	3.79	1.49	-68.3	7.6	403.9	26.6	5	4	4.5
		CCSM4_r2i1p1	7.01	1.40	-68.3	0.7	274.8	26.4	4	2	3
	Cold, wet	bcc-csm1-1_r1i1p1	16.00	1.40	-68.6	-8.6	295.7	46.8	2	5	3.5
		IPSL-CM5B-LR_r1i1p1	22.21	1.04	-75.2	-10.8	193.8	22.5	4	1	2.5
		MRI-CGCM3_r1i1p1	12.03	1.28	-87.3	-8.3	224.0	39.2	5	4	4.5
		CESM1-BGC_r1i1p1	9.75	1.36	-66	-3.9	326.1	24.0	1	2	1.5
		GISS-E2-R_r6i1p3	19.58	1.65	-73.5	-6.7	412.8	32.8	3	3	3
	Warm, Dry	MIROC-ESM-CHEM_r1i1p1	-1.22	2.16	-97.1	-4.7	643.5	1.0	5	1	3
		CMCC-CMS_r1i1p1	-1.70	2.19	-93.5	23.4	322.4	-0.3	1	5	3
		MPI-ESM-LR_r3i1p1	-6.77	2.15	-85.7	14.3	515.9	-1.4	4	2	3
		MPI-ESM-LR_r1i1p1	-7.98	2.22	-94.1	20.5	513.0	2.2	3	4	3.5
		MPI-ESM-LR_r2i1p1	-7.16	2.30	-91.8	20.1	444.5	-3.1	2	3	2.5
	Warm, wet	CanESM2_r1i1p1	14.77	2.12	-91.3	8.7	226.2	57.1	1	4	2.5
		CanESM2_r2i1p1	14.48	2.12	-89.0	-2.9	306.3	38.7	4	2	3
		CanESM2_r3i1p1	13.51	2.28	-95.3	6.5	263.4	58.4	3	5	4
		CanESM2_r5i1p1	12.27	2.11	-92.0	-9.8	236.5	40.3	2	3	2.5
		CSIRO-Mk3-6-0_r2i1p1	10.38	2.04	-88.0	-4.7	597.8	25.4	5	1	3
RCP8.5	Cold, Dry	NOAA_GFDL_GFDL-ESM2G_r1i1p1	3.40	1.91	-49.0	-2.0	372.6	21.5	2	1	1.5
		EC-EARTH_r9i1p1	4.70	1.82	-84.7	-2.5	694.9	29.0	4	2	3
		inmcm4_r1i1p1	4.15	1.53	-24.2	11.4	182.1	1.9	1	3	2
		EC-EARTH_r2i1p1	5.29	1.91	-90.5	1.9	764.3	45.1	5	5	5
		NOAA_GFDL_GFDL-ESM2M_r1i1p1	5.62	1.89	-52.4	4.3	308.2	19.4	3	4	3.5
	Cold, wet	bcc-csm1-1_r1i1p1	20.01	1.83	-68.6	-8.6	295.7	46.8	2	5	3.5
		CESM1-BGC_r1i1p1	9.22	1.93	-84.0	-3.2	448.5	29.0	1	4	2.5
		CNRM-CM5_r1i1p1	11.59	2.14	-88.6	-0.1	341.5	2.0	3	3	3
		CSIRO-Mk3-6-0_r7i1p1	12.41	2.28	-91.7	3.3	625.5	12.8	5	1	3
	Warm, Dry	CSIRO-Mk3-6-0_r1i1p1	14.23	2.29	-91.1	6.6	721.2	-0.3	4	2	3
		CMCC-CMS_r1i1p1	-2.60	3.07	-93.5	23.4	322.4	-0.3	1	2	1.5
		MIROC-ESM-CHEM_r1i1p1	0.96	3.13	-97.1	-4.7	643.5	1.0	5	5	5
		MPI-ESM-LR_r3i1p1	-1.72	2.86	-85.7	14.3	515.9	-1.4	4	1	2.5
		MPI-ESM-LR_r2i1p1	-11.69	3.06	-91.8	20.1	444.5	-3.1	3	3	3
		MPI-ESM-LR_r1i1p1	-4.49	2.73	-94.1	20.5	513.0	2.2	2	4	3
	Warm, wet	CanESM2_r2i1p1	15.68	2.92	-89.0	-2.9	306.3	38.7	4	2	3
		CanESM2_r5i1p1	19.26	2.97	-92.0	-9.8	236.5	40.3	1	5	3
		CanESM2_r1i1p1	16.70	2.91	-91.3	8.7	226.2	57.1	2	4	3
		CanESM2_r3i1p1	13.26	3.06	-95.3	6.5	263.4	58.4	3	3	3
		CSIRO-Mk3-6-0_r10i1p1	12.74	2.59	-90.5	5.6	826.6	37.4	5	1	3

source: MoFE 2019

Step 3: Final selection based on the model's skill in simulating the annual cycle of air temperature and precipitation

In this step, the final model selection is done by calculating the seasonal bias of all models for the reference period, and then comparing it with the reference dataset for the same period. Here, HI-AWARE (Himalayan Adaptation, Water and Resilience) dataset has been used as a reference dataset (Lutz & Immerzeel, 2015). For precipitation sum, the bias between the GCM run and the reference dataset is calculated on an annual basis and for the monsoon season (June–September). As about 80% of the precipitation falls during the monsoon season in Nepal, the monsoon bias will outweigh other biases originating in other periods. For the mean air temperature, the annual, summer and winter biases are used. The biases for precipitation are expressed as a percentage (relative changes) and the biases for air temperature are expressed as degree Celsius (absolute changes). However, the bias should be appropriately selected with the region where it is being applied in the mind. The bias can be calculating using following code.

```
--------------------------------------------Code begins--------------------------------------------
rm(list = ls())
# provide path to input folder with downloaded files from previous step
input_folder <- "C\\Input\\"
input_gcm <- "C\\GCM\\"
# provide path to output folder
output_folder <- "C:\\Output\\"
# set working directory
setwd(input_folder)

# list RCPs
rcps <- c('rcp45','rcp85')

#list variables
vars <- c('pr','tas')

#reference monthly dataset
p_ref <- read.table(paste(input_folder,"P_mon_ref.csv",sep=""),sep=",",header=F)

t_ref <- read.table(paste(input_folder,"T_mon_ref.csv",sep=""),sep=",",header=F)

#days in the month of a year
m_day <- as.matrix(c(31,28,31,30,31,30,31,31,30,31,30,31))

for (rcp in rcps)
{
  #reading input files according to the RCPs
 filelist <- list.files(path = input_gcm, pattern = rcp, full.names = T)}
 for (var in vars)
{
  summary <- data.frame(RCP=character(1),Model=character(1),bias_winter=numeric(1),
  bias_monsoon=numeric(1), bias_total=numeric(1),pearson=numeric(1), stringsAsFactors=FALSE)
  b <- filelist[grep(var,filelist)]
for (i in 1:length(b))
{
  infile <- b[i]

  # read the second line of the data to store model name, RCP and ensemble
  # check for the number of lines to be skipped
  modelline <- scan(infile[i], '', skip = 1, nlines = 34, sep = '\n')
  line <- modelline[grep("operating",modelline)]
  linesplit <- unlist(strsplit(line,split=" "))

  # store the model name
  model <- linesplit[4]

  # store the ensemble
  ensemble <- linesplit[11]
```

```
  # read area averaged monthly data in to a data frame
  data = read.table(infile, sep="",col.names=c("year","Jan","Feb","Mar","Apr","May","Jun",
  "Jul","Aug","Sep","Oct","Nov","Dec"))

  # taking GCM data for the historical period
  com_data <- data[data$year>1980&data$year<2011,]

  #creating empty vector to store data
  avg_gcm <- c()
  avg_ref <- c()

  for(z in 1:12)
  {
    # For GCM datasets
    if(var=="pr")
    {
      # calculating monthly sum for precipitation
      avg_gcm[z] <- mean(com_data[,z+1])*m_day[z,1]
    } else {
      # calculating monthly average for temperature
      avg_gcm[z] <- mean(com_data[,z+1])
    }
    # For Reference dataset
  if(var=="pr")
  {
    # calculating monthly sum for precipitation
    avg_ref[z] <- mean(p_ref[,z+1])*m_day[z,1]
  } else {
    # calculating monthly average for temperature
    avg_ref[z] <- mean(t_ref[,z+1])
  }
  }
  # saving data as data frame
  x <- as.data.frame(avg_ref)
  y <- as.data.frame(avg_gcm)

  #calculating seasonal bias (relative change for Precipitation and absolute change for Temperature)
  if(var=="pr")
  {
    bias_winter <- ((sum(y[1:2,1],y[12,1])-sum(x[1:2,1],x[12,1]))/sum(x[1:2,1],x[12,1]))*100
    bias_monsoon <- ((sum(y[6:9,1])-sum(x[6:9,1]))/sum(x[6:9,1]))*100
    bias_total <- ((sum(y[1:12,1])-sum(x[1:12,1]))/sum(x[1:12,1]))*100
  } else {
    bias_winter <- mean(y[1:2,1],y[12,1])-mean(x[1:2,1],x[12,1])
    bias_monsoon <- mean(y[6:9,1])-mean(x[6:9,1])
    bias_total <- mean(y[1:12,1])-mean(x[1:12,1])
  };

  # co-efficent of correlation calculation between the historical period of GCM and reference dataset
  corr <- cor(x,y,method="pearson")

  #storing the calculated bias in an array
  row <- c(rcp,paste(model,ensemble, sep="_"),bias_winter,bias_monsoon,bias_total,corr)

  # combining the data for all GCMs
  summary <- rbind(summary,row)
}

# writing the calculated bais into a csv file
outfile <- paste(output_folder,var,"_",rcp,"_bias.csv",sep="")
write.csv(summary,outfile)

}
print("finished")
}
----------------------------------------------Code ends----------------------------------------------
```

After calculating the biases for the selected models from Step 2, the values are normalized (each bias value expressed as a fraction of the largest bias value) within the ensemble for both RCP4.5 and 8.5 separately. The bias score for each model is then calculated by averaging the P bias and T bias scores (Table 4). Finally, a combined score is calculated by summing up the resulting two values. Four models with the lowest combined bias score are thus chosen each for RCP4.5 and RCP8.5. In this way, 4 models representing 4 corners of the spectrum of projections for RCP4.5 and RCP8.5 are chosen (highlighted in green).

TABLE 4 BIASES BETWEEN GCM RUNS (2036–65) AND REFERENCE CLIMATE DATASET (1981–2010) FOR NEPAL

RCP	Projection	model	P bias total (%)	P bias monsoon (%)	T bias total (°C)	T bias monsoon (°C)	T bias winter (°C)	P bias total normalized	P bias monsoon normalized	T bias total normalized	T bias monsoon normalized	T bias winter normalized	P bias score	T bias score	Combined score
RCP4.5	Cold, Dry	NOAA_GFDL_GFDL-ESM2M_r1i1p1	11.3	12.2	0.0	0.7	-0.3	0.20	0.21	0.01	0.25	0.05	0.20	0.08	0.28
		inmcm4_r1i1p1	-3.7	-13.9	-2.3	-0.7	-2.8	0.07	0.24	0.63	0.24	0.57	0.15	0.52	0.67
		CCSM4_r1i1p1	21.5	17.1	-1.1	0.0	-2.9	0.38	0.29	0.30	0.00	0.60	0.34	0.30	0.63
		CCSM4_r2i1p1	21.4	20.5	-1.0	0.0	-2.6	0.38	0.35	0.26	0.01	0.53	0.37	0.26	0.63
	Cold, wet	bcc-csm1-1_r1i1p1	-30.9	-48.1	-0.6	1.2	-3.4	0.55	0.81	0.16	0.43	0.69	0.68	0.36	1.05
		MRI-CGCM3_r1i1p1	-53.0	-58.7	-0.5	0.5	-2.2	0.95	0.99	0.13	0.17	0.44	0.97	0.22	1.19
	Warm, Dry	MIROC-ESM-CHEM_r1i1p1	6.7	-14.8	0.4	0.6	-0.2	0.12	0.25	0.12	0.19	0.04	0.18	0.12	0.30
		CMCC-CMS_r1i1p1	-0.5	-10.0	-0.3	1.0	-2.3	0.01	0.17	0.07	0.34	0.47	0.09	0.24	0.33
		MPI-ESM-LR_r3i1p1	11.2	12.4	-1.3	-2.9	-4.7	0.20	0.21	0.89	1.00	0.95	0.21	0.93	1.14
		MPI-ESM-LR_r1i1p1	14.4	12.9	0.1	0.4	-0.9	0.26	0.22	0.03	0.14	0.18	0.24	0.09	0.33
	Warm, wet	CanESM2_r2i1p1	-35.7	-39.4	-3.6	-2.0	-4.9	0.64	0.67	0.98	0.69	1.00	0.65	0.91	1.57
		CanESM2_r3i1p1	-38.6	-42.6	-3.7	-2.1	-4.7	0.69	0.72	1.00	0.73	0.96	0.70	0.92	1.62
		CSIRO-Mk3-6-0_r2i1p1	-56.0	-59.2	-0.8	1.3	-3.1	1.00	1.00	0.20	0.44	0.63	1.00	0.37	1.37
RCP8.5	Cold, Dry	EC-EARTH_r2i1p1	-12.0	-15.5	-2.8	[illegible].6	-1.9	0.22	0.27	0.70	0.98	0.36	0.24	0.68	0.92
		NOAA_GFDL_GFDL-ESM2M_r1i1p1	11.3	12.2	0.0	0.7	-0.3	0.21	0.21	0.01	0.20	0.05	0.21	0.06	0.27
	Cold, wet	bcc-csm1-1_r1i1p1	-30.9	-48.1	-0.6	1.2	-3.4	0.57	0.83	0.15	0.34	0.64	0.70	0.32	1.02
		CNRM-CM5_r1i1p1	-18.3	-25.2	-4.1	-2.1	-5.4	0.34	0.43	1.00	0.57	1.00	0.38	0.89	1.28
		CSIRO-Mk3-6-0_r7i1p1	-54.5	-58.2	-0.8	1.2	-2.8	1.00	1.00	0.19	0.32	0.53	1.00	0.31	1.31
		CSIRO-Mk3-6-0_r1i1p1	-52.2	-54.3	-0.8	1.3	-3.1	0.96	0.93	0.19	0.35	0.58	0.95	0.32	1.27
	Warm, Dry	MIROC-ESM-CHEM_r1i1p1	6.7	-14.8	0.4	0.6	-0.2	0.12	0.25	0.11	0.15	0.04	0.19	0.10	0.29
		MPI-ESM-LR_r2i1p1	18.2	17.6	-2.9	-2.7	-3.7	0.33	0.30	0.70	0.74	0.69	0.32	0.71	1.03
		MPI-ESM-LR_r1i1p1	14.4	12.9	0.1	0.4	-0.9	0.26	0.22	0.02	0.11	0.16	0.24	0.08	0.32
	Warm, wet	CanESM2_r2i1p1	-35.7	-39.4	-3.6	-2.0	-4.9	0.66	0.68	0.90	0.54	0.92	0.67	0.81	1.48
		CanESM2_r5i1p1	-38.0	-40.5	-3.5	-1.8	-4.7	0.70	0.70	0.87	0.49	0.87	0.70	0.77	1.47
		CanESM2_r1i1p1	-37.8	-40.7	-3.5	-1.9	-4.7	0.69	0.70	0.87	0.52	0.88	0.70	0.78	1.48
		CanESM2_r3i1p1	-38.6	-42.6	-3.7	-2.1	-4.7	0.71	0.73	0.91	0.57	0.88	0.72	0.82	1.54
		CSIRO-Mk3-6-0_r10i1p1	-54.4	-57.9	-2.9	-3.7	-2.2	1.00	0.99	0.71	1.00	0.40	1.00	0.71	1.70

Source: MoFE, 2019

Step 4: Downscaling of selected GCMs using quantile mapping

Step 4.1: Pre-processing of the GCMs

Before downscaling the GCM, we need to pre-process the datasets. This is done in three parts.

STEP 4.1.1: CROPPING THE GCM DATASETS

The spatial extent of most GCM datasets covers the whole world, while the temporal extent of these datasets covers both the past (historical) and future periods. The future period might be further divided into many parts. The file size of these datasets for the whole period and extent is rather large. So, we need to crop these datasets to suit our study area. Thus, the file size will be manageable for further processing. The GCM dataset can be cropped using following code

```
---------------------------------------------Code begins---------------------------------------------
rm(list=ls())
## cropping the historical dataset
# load the required packages to use the functions for the code to work
# Packages can be installed using "install.packages()" command
library(raster)
library(ncdf4)
library(RNetCDF)

rcp_in_folder <- "C:\\GCM_Historical\\"
rcp_out_folder <- "C:\\RCP4.5_clip\\"

# set working directory to the input folder
setwd(rcp_in_folder)
vars <- c('pr','tas_','tasmax','tasmin')

# list files in working directory
fl <- list.files()

# latitude and longitude of area of study (Nepal's boundary given as example)
lon_s_value <- 79
lon_e_value <- 91
lat_s_value <- 26
lat_e_value <- 31

for (var in vars)
{
  filelist <- fl[grep(var,fl)]
  for (j in 1:length(filelist))
  {
    if(var=='tas_'){var='tas'}

    # Reading the input file
    ncFile <- nc_open(filelist[j])

    # masking the study area from the GCM dataset
    LonIdx <- which(ncFile$dim$lon$vals >= lon_s_value & ncFile$dim$lon$vals <= lon_e_value)
    LatIdx <- which(ncFile$dim$lat$vals >= lat_s_value & ncFile$dim$lat$vals <= lat_e_value)
    TimIdx <- seq(1:length(ncFile$dim$time$vals))

    # extracting the data for the study area
    MyVariable <- ncvar_get(ncFile,var)[LonIdx,LatIdx,TimIdx]

    ## Write data into new netcdf file
    # define the dimensions of the array to store the extracted data
    lon_start <- ncFile$dim$lon$vals[LonIdx[1]]
    lon_end <- ncFile$dim$lon$vals[LonIdx[length(LonIdx)]]
    lon_interval <- (lon_end-lon_start)/(length(LonIdx)-1)
    lat_start <- ncFile$dim$lat$vals[LatIdx[1]]
    lat_end <- ncFile$dim$lat$vals[LatIdx[length(LatIdx)]]
    lat_interval <- (lat_end-lat_start)/(length(LatIdx)-1)
    time_start <- ncFile$dim$time$vals[TimIdx[1]]
```

```
    time_end <- ncFile$dim$time$vals[TimIdx[length(TimIdx)]]

    # define the units of dimensions
    x <- ncdim_def( "lon", "degrees_east", seq(lon_start,lon_end,lon_interval),longname = "Longitude")
    y <- ncdim_def( "lat", "degrees_north",seq(lat_start,lat_end,lat_interval),longname = "Latitude")

    # calendar and "days since" should match with the input file
    t <- ncdim_def( "time", "days since 1850-01-01 00:00:00",
                    seq(time_start,time_end), unlim=TRUE,longname = "time",calendar = "noleap")

    # define missing value
    mv <- 1e+20

    # define the variable to be stored in the netcdf file
    if(var=="pr"){var_pr <- ncvar_def('pr','kg m-2 s-1',list(x,y,t),mv,longname = 'Precipitation',
prec="float")}
    if(var=='tas'){var_pr <- ncvar_def('tas','K',list(x,y,t),mv,longname =
'Near-Surface Air Temperature',prec="float")}
    if(var=='tasmax'){var_pr <- ncvar_def('tasmax','K',list(x,y,t),mv,longname =
'Daily Maximum Near-Surface Air Temperature',prec="float")}
    if(var=='tasmin'){var_pr <- ncvar_def('tasmin','K',list(x,y,t),mv,longname =
'Daily Minimum Near-Surface Air Temperature',prec="float")}

    # create new netCDF file to store the data
    nc_pr <- nc_create(paste(rcp_out_folder,"clip_",filelist[j],sep = ""),list(var_pr), force_v4=F,
verbose=FALSE)

    # store the data into the netCDF file
    ncvar_put(nc_pr,var_pr,MyVariable)

    # close the netCDF file
    nc_close(nc_pr)
  }
}
# clear the memory used by R and restart R
gc()
.rs.restartR()
----------------------------------------------Code ends---------------------------------------------
```

The future RCP4.5 and RCP8.5 datasets are also cropped using the same code. The main consideration is to match the time variable of the dataset with the corresponding historical dataset.

```
---------------------------------------------Code begins--------------------------------------------
rm(list=ls())
## cropping the future dataset ##
library(raster)
library(ncdf4)
library(RNetCDF)

rcp_in_folder <- "C:\\RCP4.5\\"
rcp_out_folder <- "C:\\RCP4.5_clip\\"
vars <- c('pr','tas_','tasmax','tasmin')

setwd(rcp_in_folder)

# latitude and longitude of area of study (Nepal's boundary given as example)
lon_s_value <- 79
lon_e_value <- 91
lat_s_value <- 26
lat_e_value <- 31
# no. of days to be added to the rcp file to match with the historical dataset(Eg: 01011850 to
01012006) according to calendar type (this might not be always required)
ndays <- 56940

fl <- list.files()
```

```
for (var in vars)
# list of files in the working folder
  filelist <- fl[grep(var,fl)]
  for (j in 1:length(filelist))
  {
    if(var=='tas_'){var='tas'}

    # Reading the input file
    ncFile <- nc_open(filelist[j])

    # masking the study area from the GCM dataset
    LonIdx <- which(ncFile$dim$lon$vals >= lon_s_value & ncFile$dim$lon$vals <= lon_e_value)
    LatIdx <- which(ncFile$dim$lat$vals >= lat_s_value & ncFile$dim$lat$vals <= lat_e_value)
    TimIdx <- seq(1:length(ncFile$dim$time$vals))

    # extracting the data for the study area
    MyVariable <- ncvar_get(ncFile,var)[LonIdx,LatIdx,TimIdx]

    ## Write data into new netcdf file

    # define the dimensions of the array to store the extracted data
    lon_start <- ncFile$dim$lon$vals[LonIdx[1]]
    lon_end <- ncFile$dim$lon$vals[LonIdx[length(LonIdx)]]
    lon_interval <- (lon_end-lon_start)/(length(LonIdx)-1)
    lat_start <- ncFile$dim$lat$vals[LatIdx[1]]
    lat_end <- ncFile$dim$lat$vals[LatIdx[length(LatIdx)]]
    lat_interval <- (lat_end-lat_start)/(length(LatIdx)-1)

    #adding no. of days to match with the historical dataset(01011850 to 01012006) according to calendar type
    time_start <- ncFile$dim$time$vals[TimIdx[1]] + ndays
    time_end <- ncFile$dim$time$vals[TimIdx[length(TimIdx)]] + ndays

    # define the units of dimensions
    x <- ncdim_def( "lon", "degrees_east", seq(lon_start,lon_end,lon_interval),longname = "Longitude")
    y <- ncdim_def( "lat", "degrees_north",seq(lat_start,lat_end,lat_interval),longname = "Latitude")

    # calendar and "days since" should match with the input file
    t <- ncdim_def( "time", "days since 1850-01-01 00:00:00",
                seq(time_start,time_end), unlim=TRUE,longname = "time",calendar = "noleap")

    # define missing value
    mv <- 1e+20

    # define the variable to be stored in the netcdf file
    if(var=="pr"){var_pr <- ncvar_def('pr','kg m-2 s-1',list(x,y,t),mv,longname =
'Precipitation',prec="float")}
    if(var=='tas'){var_pr <- ncvar_def('tas','K',list(x,y,t),mv,longname =
'Near-Surface Air Temperature',prec="float")}
    if(var=='tasmax'){var_pr <- ncvar_def('tasmax','K',list(x,y,t),mv,longname =
'Daily Maximum Near-Surface Air Temperature',prec="float")}
    if(var=='tasmin'){var_pr <- ncvar_def('tasmin','K',list(x,y,t),mv,longname =
'Daily Minimum Near-Surface Air Temperature',prec="float")}

    # create new netCDF file to store the data
    nc_pr <- nc_create(paste(rcp_out_folder,"clip_",filelist[j],sep = ""),list(var_pr), force_v4=F,
verbose=FALSE)

    # store the data into the netCDF file
    ncvar_put(nc_pr,var_pr,MyVariable)

    # close the netCDF file
    nc_close(nc_pr)
  }
}
# clear the memory used by R to store the data then restart R
gc()
.rs.restartR()
----------------------------------------------Code ends----------------------------------------------
```

STEP 4.1.2: SELECTING AND MERGING THE TIME FRAME (1981–2010) FROM THE GCM DATASETS

After we have cropped the GCM datasets to suit our study area, we need to merge the historical and future datasets into a single file. We also need to select the appropriate period to downscale the dataset. In this case, the time frame used was of 1981–2100. We use Climate Data Operator (CDO) to select the appropriate year and merge the historical and future datasets.

```
---------------------------------------------Code begins---------------------------------------------
rm(list=ls())
library(stringr)

#output folder from previous step will be input folder for this step
input_folder <- “C:\\rcp_out_folder\\”
output_folder <- “C:\\rcp_merged\\”
working_folder <- “C:\\temp\\”
abc <- list.files(input_folder)

# storing the file names without extensions
split1 <- str_sub(abc[1],start = 12L,end = -21L)

vars <- c(‘pr’,’tas_’,’tasmax’,’tasmin’)

for (var in vars)
{
  if(var==’tas_’){var=’tas’}
  # sel year selects the given year from the input file
  command1 <- paste(“cdo selyear,1981/2005 “,input_folder,”\\clip_”,var,split1,”19750101-20121231.nc
“,working_folder,”\\clip_”,var,split1,”19810101-20051231.nc”,sep =”” )

  # prints the string (used here to check progress)
  print(command1)

  # system command passes the command to command prompt
  system(command1)
}

# list of files in the working folder
filelist <- list.files(working_folder)

# setting the working directory
setwd(working_folder)

for (var in vars)
{
    # listing all files for a GCM to be merged into one
  file1 <- filelist[grep(var, filelist)]
  fl <- paste(file1,collapse = “ ”)

  if(var==’tas_’){var=’tas’}
  # mergetime merges all the input file into one
  command1 <- paste(“cdo mergetime “, fl,” “,var,”_day_EC-EARTH_rcp85_r2i1p1_1981-2100.nc”,sep = “”)
  print(command1)
  system(command1)
}
----------------------------------------------Code ends----------------------------------------------
```

STEP 4.1.3: PROJECTING THE GCM DATASETS FROM GEOGRAPHIC (WGS 1984) TO THE PROJECTED (UTM 45N) COORDINATE SYSTEM

The last step of pre-processing is to project the merged GCM datasets from the World Geodetic System (WGS) 1984 Geographic Projection System (degree decimal) on to the Universal Transverse Mercator (UTM) Zone 45N Projected Coordinate System (metres). We need the projected dataset in order to downscale the GCM datasets. We will also mask and interpolate the projected GCM dataset with the extent of our reference dataset. This will reduce the downscaling computation time by eliminating the cell outside the extent/boundary of our reference dataset.

```
---------------------------------------------Code begins---------------------------------------------
## script written by Saurav Pradhananga

rm(list=ls())

library(raster)

input_folder <- “C:\\rcp_merged\\”
output_folder <- “C:\\rcp_projected\\”
working_folder <- “C:\\temp\\”

filelist1 <- list.files(input_folder,full.names = T)

vars <- c(‘pr’,’tas_’,’tasmax’,’tasmin’)

# list of all GCM to be downscaled
GCMs <- c(“bcc-csm1-1_rcp45_r1i1p1”,”bcc-csm1-1_rcp85_r1i1p1”,”GFDL-ESM2M_rcp45_r1i1p1”,”CanESM2_
rcp45_r2i1p1”, “MIROC-ESM-CHEM_rcp45_r1i1p1”,”CanESM2_rcp85_r5i1p1”,”GFDL-ESM2M_rcp85_r1i1p1”, “MI-
ROC-ESM-CHEM_rcp85_r1i1p1”)

# reference time period
startyear <- 1981
endyear <- 2010

# masking layer of reference dataset
clone <- raster(“C:\\temp\\temp.tif”)

# projection parameters for UTM45N
projection_param <- “+proj=utm +zone=45 +datum=WGS84 +units=m +no_defs +ellps=WGS84 +towgs84=0,0,0”

for (GCM in GCMs)
{
   dates<-seq(as.Date(“1981-01-01”),as.Date(“2010-12-31”),”day”)
   timesteps <- 1:length(dates)

   # dataframe to store the dates of reference period
   datesframe <- matrix(data=NA,nrow=length(timesteps),ncol=4)
   datesframe[,1] <- timesteps
   for (i in timesteps)
   {
     datesframe[i,2] <- as.numeric(format(dates[i], “%Y”))
     datesframe[i,3] <- as.numeric(format(dates[i], “%m”))
     datesframe[i,4] <- as.numeric(format(dates[i], “%d”))
   }

   # removing leap days from the date dataframe as most of the GCM dataset lack leap days
   leapdays <- datesframe[which(datesframe[,3]==2 & datesframe[,4]==29)]

for (var in vars)
{
```

```
  if(var=="tas_"){var="tas"}
  infile <- paste(input_folder,var,"_day_",GCM,"_19810101-21001231.nc",sep="")
  # extract reference period from infile
  timsel <- paste(working_folder,var,"_",GCM,"_",startyear,"_",endyear,".nc",sep="")
  print(paste("Extract reference period for ",GCM," for ",var," inputfile...",sep=""))
  command <- paste("cdo selyear,",startyear,"/",endyear," ",infile," ",timsel,sep="")
  system(command)
  # read GCM data as RasterBrick and project to UTM and interpolate to clone resolution (bilinear)
  print(paste("Read ",GCM," ",var," data as RasterBrick...",sep=""))
  # store the dataset as raster stack
  RB <- brick(timsel)

  # project the raster to UTM
  projected_RB <- projectRaster(RB,crs = projection_param)
  clipped_RB <- crop(projected_RB,clone,snap="out")
  RB_proj <- projectRaster(clipped_RB,clone,method="bilinear")

  # remove the layer that is not required for further analysis to free up the memory
  rm(RB)

  print(paste("Convert units and mask with clone for ",GCM," for ",var,"...",sep=""))
  if(var=='pr')
  {
    # converting the units of precipitation from kgm-1s-1 to mm
    RB_proj_conv <- RB_proj * 86400 * clone
    var_unit <- "mm"
  }
  if (var=='tas' | var=='tasmax' | var=='tasmin')
  {
    # converting the units of temperture from Kelvin to Celsius
    RB_proj_conv <- (RB_proj - 273.15) * clone
    var_unit <- "degree_Celsius"
  }

  # remove the layer that is not required for further analysis to free up the memory
  rm(RB_proj)
  gc()

  # remove leap days for GCM runs with standard calendar
  if(GCM == 'MIROC-ESM-CHEM_rcp45_r1i1p1' | GCM == 'MIROC-ESM-CHEM_rcp85_r1i1p1')
  {
    RB_proj_conv <- dropLayer(RB_proj_conv,leapdays)
  }


  # write projected data as NetCDF
  print(paste("Write data for ",GCM," for ",var," as NetCDF...",sep=""))
  writeRaster(RB_proj_conv,filename=paste(output_folder,var,"_",GCM,"_",startyear,"_",endyear,
  "_prj.nc",sep=""),format="CDF",overwrite=T,varname=var,varunit=var_unit,zname="time",zunit="days
  since 1981-01-01 00:00:00")

  # remove the layer that is not required for further analysis to free up the memory
  rm(RB_proj_conv)
  gc()
}
}
----------------------------------------------Code ends----------------------------------------------
```

Step 4.2: Generating empirical cumulative distribution functions

The Empirical cumulative distribution function (ECDF) is a non-parametric estimator of the underlying CDF of a given variable. It orders the data (n, number of data) from the smallest to the largest value and assigns a probability of 1/n to each datum up to and including that datum. Here, we generate the ECDFs for the reference period of both observed and GCM datasets.

```
---------------------------------------------Code begins---------------------------------------------
## script written by Arthurlutz
## Modified by SauravPradhananga
rm(list=ls(all=TRUE))

library(raster)


input_folder_obs <- "C:\\Reference\\"
input_folder_gcm <- "C:\\GCM\\"
output_folder <- "C:\\ECDFs\\"

vars <- c('pr','tas','tasmax','tasmin')

GCMs <- c("bcc-csm1-1_rcp45_r1i1p1","bcc-csm1-1_rcp85_r1i1p1","GFDL-ESM2M_rcp45_r1i1p1","CanESM2_
rcp45_r2i1p1", "MIROC-ESM-CHEM_rcp45_r1i1p1","CanESM2_rcp85_r5i1p1","GFDL-ESM2M_rcp85_r1i1p1",
"MIROC-ESM-CHEM_rcp85_r1i1p1")

# coupling dates of refrerence period to process the timesteps
dates<-seq(as.Date("1981-01-01"),as.Date("2010-12-31"),"day")

#removing leap days
dates<- dates[-which(substr(dates,6,10)=="02-29")]
timesteps <- 1:10950
datesframe <- matrix(data=NA,nrow=10950,ncol=4)
datesframe[,1] <- timesteps
for (i in timesteps)
{
  datesframe[i,2] <- as.numeric(format(dates[i], "%Y"))
  datesframe[i,3] <- as.numeric(format(dates[i], "%m"))
  datesframe[i,4] <- as.numeric(format(dates[i], "%d"))
}

# calculate ecdfs for reference data
for (var in vars)
{
# create rasterbrick of observed data
if(var == 'pr'){varname <- "prec"}
if(var == 'tas'){varname <- "tavg"}
if(var == 'tasmax'){varname <- "tmax"}
if(var == 'tasmin'){varname <- "tmin"}

infile_obs <- paste(input_folder_obs,varname,"_1981-2010.nc",sep="")
OBS_b <- brick(infile_obs)

# drop leap days from observed dataset
dates1<-seq(as.Date("1981-01-01"),as.Date("2010-12-31"),"day")
timesteps1 <- 1:length(dates1)
datesframe1 <- matrix(data=NA,nrow=length(timesteps1),ncol=4)
datesframe1[,1] <- timesteps1
for (i in timesteps)
{
  datesframe1[i,2] <- as.numeric(format(dates[i], "%Y"))
  datesframe1[i,3] <- as.numeric(format(dates[i], "%m"))
  datesframe1[i,4] <- as.numeric(format(dates[i], "%d"))
}
```

```
leapdays <- datesframe1[which(datesframe1[,3]==2 & datesframe1[,4]==29)]

OBS_b <- dropLayer(OBS_b ,leapdays)

# create array of reference dataset
# loop over months
for (month in 1:12)
{
  # select timesteps for particular month and store in array
  dates_m<-datesframe[which(datesframe[,3] == month),]
  timesteps <- as.vector(dates_m[,1])
  OBS_m <- subset(OBS_b,subset=timesteps)
  OBS_m_array <- as.array(OBS_m)

  print(paste("Array for obs data for ",var," for month ",month," created...",sep=""))

  # calculcate ecdfs and store ecdfs in list
  ecdf_obs_list <- c()
  for (r in 1:nrow(OBS_m_array))
  {
    for(c in 1:ncol(OBS_m_array))
    {
      if(all(is.na(OBS_m_array[r,c,])))
      {
        ecdf_obs_list <- c(ecdf_obs_list,NA)
      }
      else
      {
        ecdf_obs_list <- c(ecdf_obs_list,list(ecdf(OBS_m_array[r,c,])))
      }
    }
  }
  # save list with ecdfs for particular month, then remove array and list of ecdfs
  save(ecdf_obs_list,file=paste(output_folder,"obs_refs\\ecdfs_",var,"_obs_1981_2010_",month,".
RData",sep=""))
  rm(OBS_m_array,OBS_m,ecdf_obs_list)
  gc()
  print(paste("ECDFs for observed data reference period for ",var," for month ",month,"
saved...",sep=""))
}
# remove raster brick that is not required for further analysis
rm(OBS_b)
gc()
}
print("Finished calculating ECDFs for observed data reference period...")


## calculate ecdfs for GCM data
# create arrays of GCM data for reference period
for (GCM in GCMs)
{
  for(var in vars)
  {
    # create rasterbrick of GCM data
    GCM_b <- brick(paste(input_folder_gcm,var,"_",GCM,"_1981_2010_prj.nc",sep=""))

    for (month in 1:12)
    {
      # select timesteps for particular month and save as array
      dates_m <- datesframe[which(datesframe[,3] == month),]
      timesteps <- as.vector(dates_m[,1])
      GCM_m <- subset(GCM_b,subset=timesteps)
      GCM_m_array <- as.array(GCM_m)
      print(paste("Array for ",GCM," reference period for ",var," for month",month," created...",sep=""))
      # calculcate ecdfs and store ecdfs in list
```

```
        ecdf_gcm_list <- c()
        for (r in 1:nrow(GCM_m_array))
        {
          for(c in 1:ncol(GCM_m_array))
          {
            if(all(is.na(GCM_m_array[r,c,])))
            {
              ecdf_gcm_list <- c(ecdf_gcm_list,NA)
            }
            else
            {
              ecdf_gcm_list <- c(ecdf_gcm_list,list(ecdf(GCM_m_array[r,c,])))
            }
          }
        }
        # save list with ecdfs for particular month, then remove array and list of ecdfs
        save(ecdf_gcm_list,file=paste(output_folder,"gcm_refs\\ecdfs_",var,"_",GC-
M,"_1981_2010_",month,".RData",sep=""))
        rm(GCM_m_array,GCM_m,ecdf_gcm_list)
        gc()
        print(paste("ECDFs for ",GCM," for reference period for ",var," for month ",month,"
saved...",sep=""))
      }
      # remove raster brick
      rm(GCM_b)
      gc()
    }
}
print("Finished calculating ECDFs for GCM data reference period...")

----------------------------------------------Code ends---------------------------------------------
```

Step 4.3: Applying the correction functions

Quantile mapping uses the ECDFs to correct the GCM datasets on a daily time step (t) for each grid cell (i). The GCM data are compared with the observed reference dataset for each grid in order to estimate the bias on a monthly basis. The bias information is applied to the future dataset by calculating the correction function (CF). The CF represents the difference between the observed and the modelled inverse ECDF for the respective day of the year in the reference period at probability P. P is obtained by relating the GCM data X^{GCM} to the corresponding ECDF in the reference period. This results in the corrected time series Y^{cor} to create the bias corrected dataset.

$$Y_{t,i}^{cor} = X_{t,i}^{GCM} + CF_{t,i}$$

$$CF_{t,i} = ecdf_{month,i}^{obs,ref^{-1}}(P_{t,i}) - ecdf_{month,i}^{GCM,ref^{-1}}(P_{t,i})$$

$$P_{t,i} = ecdf_{month,i}^{GCM,ref}(X_{t,i}^{GCM})$$

Here, we have applied frequency adaptation (FA) to extend the basic QM procedure. FA is applied in order to account for a methodological problem, which occurs if the dry-day frequency in the GCM dataset ($ecdf^{GCM,ref}$) is greater than in the observations ($ecdf^{obs,ref}$)). With FA, only the fraction (ΔP_0)

$$\Delta P_0 = \frac{ecdf_{month,i}^{GCM,ref}(0) - ecdf_{month,i}^{obs,ref}(0)}{ecdf_{month,i}^{GCM,ref}(0)}$$

of such dry-day cases with probability P_0 are corrected randomly by linearly interpolating between zero precipitation and the precipitation amount of $ecdf_{month,i}^{obs,ref^{-1}}(ecdf_{month,i}^{GCM,ref}(0))$. This will reduce the wet bias in the GCM dataset.

In addition, for values of extremes that are outside the range of the reference period, corrections are made by including the constant linear extrapolation of the correction value (i.e., the difference between $ecdf^{obs,ref}$ and $ecdf^{GCM,ref}$) at the highest and lowest quantiles. In this case, the future corrected value would be calculated as follows:

$$P_{fut,cor} = max(P_{obs}) * \frac{P_{fut,GCM}}{max(P_{fut,GCM})}$$

For a detailed description on QM, FA, and extreme value extrapolation, please refer to Themeßl et al. (2011a, 2011b).

```
-------------------------------------------------Code begins-------------------------------------------------
## script written by Arthurlutz
## modified by Saurav Pradhananga

rm(list=ls())
library(raster)

# output folder from previous step will be input folder for this step
input_ecdf <- “C:\\ECDFs\\”
input_folder_gcm <- “C:\\rcp_merged\\”

output_folder <- “C:\\Downscaled\\”
working_folder <- “C:\\temp\\”

# masking layer of reference dataset
clone <- raster(“C:\\temp\\mask.tif”)

# projection parameters of “UTM45N” for downscaled datasets
projection(clone)<-”+proj=utm +zone=45 +ellps=WGS84 +datum=WGS84 +units=m +no_defs”

# projection parameters of “WGS 1984” for GCM datasets
projection_gcm <- “+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs”
var <- c(‘pr’,’tas’,’tasmax’,’tasmin’)
GCMs <- c(“bcc-csm1-1_rcp45_r1i1p1”,”bcc-csm1-1_rcp85_r1i1p1”,”GFDL-ESM2M_rcp45_r1i1p1”,”CanESM2_
rcp45_r2i1p1”, “MIROC-ESM-CHEM_rcp45_r1i1p1”,”CanESM2_rcp85_r5i1p1”,”GFDL-ESM2M_rcp85_r1i1p1”,
“MIROC-ESM-CHEM_rcp85_r1i1p1”)

# time period to downscale the GCM dataset
startyear <- 1981
endyear <- 2100

# get properties of output raster
grid_dimensions <- dim(clone)
cells <- grid_dimensions[1]*grid_dimensions[2]

for (var in vars)
{
  if(var == ‘pr’){varname <- “prec”}
  if(var == ‘tas’){varname <- “tavg”}
  if(var == ‘tasmax’){varname <- “tmax”}
  if(var == ‘tasmin’){varname <- “tmin”}

  for (GCM in GCMs)
  {
    # coupling dates in data to process to timesteps
    dates<-seq(as.Date(“1981-01-01”),as.Date(“2100-12-31”),”day”)

    # removing leap days
    dates<- dates[-which(substr(dates,6,10)==”02-29”)]
    timesteps <- 1:43800
    datesframe <- matrix(data=NA,nrow=43800,ncol=4)
    datesframe[,1] <- timesteps
    for (i in timesteps)
    {
      datesframe[i,2] <- as.numeric(format(dates[i], “%Y”))
      datesframe[i,3] <- as.numeric(format(dates[i], “%m”))
      datesframe[i,4] <- as.numeric(format(dates[i], “%d”))
    }

    # select timesteps in period startyear-endyear
    dates<-datesframe[which(datesframe[,2] >= startyear & datesframe[,2] <= endyear),]

    infile <- paste(input_folder_gcm,var,”_”,GCM,”_1981_2100.nc”,sep=””)

    # apply the correction per month, loading one set of ecdfs at a time
    for(month in 1:12)
```

```
    {
      # load ecdf for obs and gcm data reference period for the particular month
      print(paste("Loading ecdfs for reference period for observations and ",GCM," for ",var," for
month",month,"...",sep=""))
      load(paste(input_ecdf,"obs_refs\\ecdfs_",var,"_obs_1981_2010_",month,".RData",sep=""))
      load(paste(input_ecdf,"gcm_refs\\ecdfs_",var,"_",GCM,"_1981_2010_",month,".RData",sep=""))

      # make vector with list of cell values from clone
      clone_mat <- as.matrix(clone)
      clone_vec <- rep(NA, cells)
      vec_pos <- 1
      for (r in 1:grid_dimensions[1])
      {
        for(c in 1:grid_dimensions[2])
        {
          clone_vec[vec_pos] <- clone_mat[r,c]
          vec_pos <- vec_pos + 1
        }
      }

      ##make vector with list of cell addresses of non-NA cells
      cell_addresses <- rep(NA, cells)
      vec_pos <- 1
      for (r in 1:grid_dimensions[1])
      {
        for(c in 1:grid_dimensions[2])
        {
          cell_addresses[vec_pos] <- vec_pos
          vec_pos <- vec_pos + 1
        }
      }

      cell_addresses2 <- clone_vec * cell_addresses
      cells_nonNA <- cell_addresses2[which(!is.na(cell_addresses2))]
      cells_nonNA_length <- as.numeric(length(cells_nonNA))

      # extract month-constant values from ecdfs that don't need to be extracted for each specific day
      FA <- rep(NA, cells)
      max_gcm_con <- rep(NA, cells)
      min_gcm_con <- rep(NA, cells)
      max_obs_con <- rep(NA, cells)
      min_obs_con <- rep(NA, cells)

      # initialization for splitting of ecdfs in 101 intervals
      probs <- seq(0,1,by=0.01)
      ecdf_obs_mat <- matrix(NA,nrow=101,ncol=cells)
      ecdf_gcm_mat <- matrix(NA,nrow=101,ncol=cells)

      for (z in 1:length(clone_vec))
      {
        if(!is.na(clone_vec[z]))
        {
          ##make vector stating for each cell if frequency adaptation is required
          prob0mm_gcm_con <- as.numeric(ecdf_gcm_list[z][[1]](0))
          prob0mm_obs_con <- as.numeric(ecdf_obs_list[z][[1]](0))
          if(var == 'pr')
          {
            if(prob0mm_gcm_con > prob0mm_obs_con)
            {
              FA[z] <- TRUE
            }
            if(prob0mm_gcm_con <= prob0mm_obs_con)
            {
              FA[z] <- FALSE
            }
          }
```

```
      # make vector with for each cell the maximum in the gcm ecdf (at probability=0)
      # make vector with for each cell the minimum in the gcm ecdf (at probality=1)
      max_gcm_con[z] <- as.numeric(quantile(ecdf_gcm_list[z][[1]],1))
      min_gcm_con[z] <- as.numeric(quantile(ecdf_gcm_list[z][[1]],0))

      # make vector with for each cell the maximum in the obs ecdf (at probability=0)
      # make vector with for each cell the minimum in the obs ecdf (at probality=1)
      max_obs_con[z] <- as.numeric(quantile(ecdf_obs_list[z][[1]],1))
      min_obs_con[z] <- as.numeric(quantile(ecdf_obs_list[z][[1]],0))

      # split ecdfs in intervals of 1% probability
      ecdf_gcm_mat[,z] <- as.numeric(quantile(ecdf_gcm_list[[z]],probs))
      ecdf_obs_mat[,z] <- as.numeric(quantile(ecdf_obs_list[[z]],probs))

    }
  }

  # get all dates/timesteps to process for this month
  dates_sub <- dates[which(dates[,3] == month),]

  # loop over all days for this month in entire period
  for (i in 1:nrow(dates_sub))
  {
    # read raw GCM grid for particular day
    grid_raw <- raster(infile,band=dates_sub[i,1])
    projection(grid_raw) <- projection_gcm

    # reproject to clone resolution, extent, projection,convert units and mask with clone
    grid_prj1 <- projectRaster(grid_raw,clone,method=”bilinear”)

    if(var==’pr’)
    {
      # kg/m2/s to mm/day
      grid_prj <- grid_prj1 * 86400 * clone
    }

    if (var==’tas’ | var==’tasmax’ | var==’tasmin’)
    {
      # deg K to deg C
      grid_prj <- (grid_prj1 - 273.15) * clone
    }

    # convert to array
    grid_arr <- as.array(grid_prj)

    # make vector with list of cell values
    grid_vec <- rep(NA, cells)
    vec_pos <- 1
    for (r in 1:nrow(grid_arr))
    {
      for(c in 1:ncol(grid_arr))
      {
        grid_vec[vec_pos] <- grid_arr[r,c,]
        vec_pos <- vec_pos + 1
      }
    }

    # loop over vector of nonNA cell values to do the transformation
    DS <- rep(NA, cells)
    for (z in cells_nonNA)
    {
      # apply ordinary Quantile Mapping when future GCM value which is
      # within range of ecdf of GCM control run (refrence period)
      if(grid_vec[z] >= min_gcm_con[z] & grid_vec[z] <= max_gcm_con[z])
      {
```

```
      for (m in 1:101)
      {
        if(ecdf_gcm_mat[m,z] > grid_vec[z])
        {
          gcm_max <- ecdf_gcm_mat[m,z]
          gcm_min <- ecdf_gcm_mat[m-1,z]
          prob <- ((m-2)/100) + 0.01 * ((grid_vec[z] - gcm_min)/(gcm_max-gcm_min))
          break
        }
      }
      index_max <- ceiling(prob*100)
      index_min <- floor(prob*100)
      obs_min <- ecdf_obs_mat[index_min+1,z]
      obs_max <- ecdf_obs_mat[index_max+1,z]

      DS[z] <- round(obs_min + (obs_max - obs_min)*(prob*100-index_min),2)
    }


    ## new extremes (values which are higher than the maximum value during observed period)
    # linear extrapolation at highest quantile if future GCM value is larger -
    # - than maximum in ecdf of GCM control run (refrence period)
    # relative change for precipitation and absolute change for temperature
    else if(grid_vec[z] > max_gcm_con[z])
    {
      if (var == 'pr')
      {
        DS[z] <- grid_vec[z] * (max_obs_con[z]/max_gcm_con[z])
      }
      else if(var == 'tas' | var == 'tasmax' | var == 'tasmin')
      {
        DS[z] <- grid_vec[z] - (max_gcm_con[z]- max_obs_con[z])
      }
    }
    ## new extremes (values which are lower than the minimum value during observed period)
    # linear extrapolation at lowest quantile if future GCM value is smaller than minimum
    in ecdf of GCM control run (refrence period)
    else if(grid_vec[z] < min_gcm_con[z])
    {
      if(var == 'pr')
      {
        # downscaled value = 0 (for precipitation)
        DS[z] <- 0
      }
      else if(var == 'tas' | var == 'tasmax' | var == 'tasmin')
      {
        # linear extrapolation at lowest quantile if future GCM value is smaller -
        # - than minimum in ecdf of GCM control run (for temperature)
        DS[z] <- grid_vec[z] - (min_gcm_con[z]- min_obs_con[z])
      }
    }

    # frequency adaptation for precipitation(if probality of 0 mm precipitation is larger in
    gcm ecdf than in obs ecdf)
    # only when precipitation according to gcm future is 0 mm (dry day), and only a fraction of
    the dry days
    # random value between 0 mm and precipiation amount at probability 0 for obs control run
    if(FA[z]==TRUE & var == 'pr' & grid_vec[z] == 0)
    {
      fraction <- (ecdf_gcm_list[z][[1]](0)-ecdf_obs_list[z][[1]](0))/ecdf_gcm_list[z][[1]](0)
      random <- runif(1,min=0,max=1)
      if(random < fraction)
      {
      DS[z] <- runif(1,min=0,max=as.numeric(quantile(ecdf_obs_list[z][[1]],ecdf_gcm_list[z]
[[1]](0))))
```

```
                print("Applied FA")
              }
            }
          }

          ##build downscaled raster from vector
          # initiate array and position iterator for vector
          pos <- 1
          DS_arr <- array(NA,dim=c(grid_dimensions[1],grid_dimensions[2],1))
          # loop over all slots in array and fill with downscaled values
          for (r in 1:nrow(DS_arr))
          {
            for(c in 1:ncol(DS_arr))
            {
              #include bottom limitation of 0 mm for precipitation
              if(var == 'pr')
              {
                DS_arr[r,c,1] <- max(0,DS[pos])
              }
              else if(var == 'tas' | var == 'tasmax' | var == 'tasmin')
              {
                DS_arr[r,c,1] <- DS[pos]
              }
              pos <- pos + 1
            }
          }

          # convert array to raster
          grid_DS <- raster(DS_arr[,,1])
          projection(grid_DS) <- projection(clone)
          extent(grid_DS) <- extent(clone)


          # write raster to GeoTIFF format
          timestep <- sprintf("%07d",dates_sub[i,1])
          file_no <- paste(substr(timestep,1,4),substr(timestep,5,7),sep="")
          outfile <- paste(output_folder,GCM,"\\",varname,"\\",varname,"_",file_no,".tif",sep="")
          writeRaster(grid_DS, outfile, format="GTiff", overwrite=TRUE)
          print(paste("Processed ",var," for ",GCM," for ",dates_sub[i,2],"-",dates_sub[i,3],"-",
          dates_sub[i,4],sep=""))
          }

        # clean up memory
        rm(ecdf_gcm_list,ecdf_obs_list)
        gc()
      }
    }
  }

----------------------------------------------Code ends---------------------------------------------
```

Step 4.4: Conversion of daily raster layer to yearly NetCDF format

The previous step will produce daily output (43,800 files) for 120 years (1981– 2100). To consolidate the output into yearly NetCDFs (Network Common Data Forms), we use the following script. Here, consideration should to taken while setting the attributes of the variable.

```
------------------------------------------------Code begins------ -----------------------------------------
## script written by Arthurlutz
## modified by Rene Wijngaard and Saurav Pradhananga
rm(list=ls())

library(RNetCDF)
library(raster)

# output folder from previous step will be input folder for this step
input_folder <- "C:\\Downscaled\\"
output_folder <- "C:\\NETCDFs\\"
vars <- c('prec','tavg','tmax','tmin')

GCMs <- c("bcc-csm1-1_rcp45_r1i1p1","bcc-csm1-1_rcp85_r1i1p1","GFDL-ESM2M_rcp45_r1i1p1","CanESM2_
rcp45_r2i1p1", "MIROC-ESM-CHEM_rcp45_r1i1p1","CanESM2_rcp85_r5i1p1","GFDL-ESM2M_rcp85_r1i1p1", "MI-
ROC-ESM-CHEM_rcp85_r1i1p1")

# dummy raster to get extent and resolution of downscaled dataset
clone <- raster("C:\\Downscaled\\prec_0000001.tif")

# time period to downscale the GCM dataset
startyear <- 1981
endyear <- 2100

# coupling dates in data to process to timesteps
dates<-seq(as.Date("1981-01-01"),as.Date("2100-12-31"),"day")

# removing leap days
dates<- dates[-which(substr(dates,6,10)=="02-29")]
timesteps <- 1:length(dates)
datesframe <- matrix(data=NA,nrow=length(dates),ncol=4)
datesframe[,1] <- timesteps
for (i in timesteps)
{
  datesframe[i,2] <- as.numeric(format(dates[i], "%Y"))
  datesframe[i,3] <- as.numeric(format(dates[i], "%m"))
  datesframe[i,4] <- as.numeric(format(dates[i], "%d"))
}

# select timesteps in period startyear-endyear
dates<-datesframe[which(datesframe[,2] >= startyear & datesframe[,2] <= endyear),]

# extract dimensions/extent/resolution/cellcenters
grid_dimensions <- dim(clone)
extent <- extent(clone)
resolution <- (extent[2]-extent[1])/grid_dimensions[2]

grid_metadata <- as.data.frame(matrix(ncol=2,nrow=grid_dimensions[1]*grid_dimensions[2]))
m<-1
for (i in 1:grid_dimensions[2]){
  for (y in 1: grid_dimensions[1])
  {
    grid_metadata[m,2] <- extent[1]+i*resolution-0.5*resolution
    grid_metadata[m,1] <- extent[3]+y*resolution-0.5*resolution
    m <- m+1
  }
}
colnames(grid_metadata) <- c("ycenter","xcenter")

# loop over variables
dir.create(paste(output_folder,"\\",GCM,sep=""))
for (GCM in GCMs)
{
  for (var in vars)
  {
```

```
##loop over years
for(year in startyear:endyear)
{
  yeardays <- datesframe[which(datesframe[,2] == year),]
  dim_yeardays <- dim(yeardays)[1]
  ##initiate array to store daily data
  Pdata <- array(NA,dim=c(grid_dimensions[2],grid_dimensions[1],dim_yeardays))
  #loop over days and fill array with data from daily grids
  for (i in 1:dim_yeardays)
  {
    timestep <- sprintf("%07d", yeardays[i,1])
    pcrno <- paste(substr(timestep,1,4),substr(timestep,5,7),sep="")
    print(paste(yeardays[i,2],"-",yeardays[i,3],"-",yeardays[i,4],sep=""))
    grid <- raster(paste(input_folder,GCM,"\\",var,"\\",var,"_",pcrno,".tif",sep=""))
    tempPdata <- t(as.matrix(grid))
    Pdata[,,i]<-tempPdata
  }


  ###################### Create NetCDF Output #########################

  # create new netcdf file ("clobber=TRUE" overwrites existing files!)
  new <- create.nc(paste(output_folder,GCM,"\\",var,"_",year,".nc",sep=""),clobber=TRUE);

  # define the dimensions
  dim.def.nc(new,dimname="latitude", dimlength=grid_dimensions[1],unlim=FALSE);
  dim.def.nc(new,dimname="longitude",dimlength=grid_dimensions[2],unlim=FALSE);
  dim.def.nc(new,dimname="time",dimlength=dim_yeardays,unlim=FALSE);


  ## define the variables and attributes
  #longitude
  var.def.nc(new,varname="longitude",vartype="NC_FLOAT",dimensions=c("longitude"));
  att.put.nc(new,variable="longitude",name="long_name",type="NC_CHAR",value="Longitude");
  att.put.nc(new,variable="longitude",name="_CoordinateAxisType",type="NC_CHAR",value="Lon");
  att.put.nc(new,variable="longitude",name="units",type="NC_CHAR",value="degrees_east");

  #latitude
  var.def.nc(new,varname="latitude",vartype="NC_FLOAT", dimensions=c("latitude"));
  att.put.nc(new,variable="latitude",name="long_name",type="NC_CHAR",value="Latitude");
  att.put.nc(new,variable="latitude",name="_CoordinateAxisType",type="NC_CHAR",value="Lat");
  att.put.nc(new,variable="latitude",name="units",type="NC_CHAR",value="degrees_north");

  #time
  var.def.nc(new,varname="time",vartype="NC_FLOAT",dimensions=c("time"));
  att.put.nc(new,variable="time",name="long_name",type="NC_CHAR",value="time");
  att.put.nc(new,variable="time",name="units",type="NC_CHAR",
             value=paste("days since ",year-1,"-12-31 12:0:0",sep=""))
  att.put.nc(new,variable="time",name="calendar",type="NC_CHAR",value="standard")

  #set attributes according to the variable
  var.def.nc(new,varname="P",vartype="NC_FLOAT",dimensions=c("longitude","latitude","time"));
  att.put.nc(new,variable="P",name="standard_name",type="NC_CHAR",value="precipitation");
  att.put.nc(new,variable="P",name="long_name",type="NC_CHAR",value="Daily precipitation sum (mm)");
  att.put.nc(new,variable="P",name="units",type="NC_CHAR",value="mm");
  att.put.nc(new,variable="P",name="_FillValue",type="NC_FLOAT",value=-9999);


  #setting projection parameters of the output file
  var.def.nc(new,varname="UTM_Projection",vartype="NC_CHAR",dimensions=NA);
  att.put.nc(new,variable="UTM_Projection",name="grid_mapping_name",type="NC_CHAR",
  value="universal_transverse_mercator")
  att.put.nc(new,variable="UTM_Projection",name="utm_zone_number",type="NC_FLOAT", value="45")
  att.put.nc(new,variable="UTM_Projection",name="semi_major_axis",type="NC_FLOAT", value="6378137")
```

```
      att.put.nc(new,variable="UTM_Projection",name="inverse_flattening",type="NC_FLOAT",
      value="298.257")
      att.put.nc(new,variable="UTM_Projection",name="_CoordinateTransformType",type="NC_CHAR",
      value="Projection")
      att.put.nc(new,variable="UTM_Projection",name="_CoordinateAxisTypes",type="NC_CHAR",
      value="GeoX GeoY")

      # close and reopen the netcdf file to enable write access
      close.nc(new)
      new <- open.nc(paste(output_folder,GCM,"\\",var,"_",year,".nc",sep=""), write=TRUE);

      # put the parameter data into the netcdf variables
      var.put.nc(new,variable="longitude",data=sort(unique(grid_metadata$xcenter)));
      var.put.nc(new,variable="latitude",data=rev(sort(unique(grid_metadata$ycenter))));
      var.put.nc(new,variable="time",data=c(1:dim_yeardays));

      # store data into the netCDF file
      var.put.nc(new,variable="P",data=Pdata);


      # Add global attributes to the NetCDF file
      att.put.nc(new, "NC_GLOBAL", "comment", "NC_CHAR", "This NetCDF file has been generated using
      the RNetCDF library in R");
      att.put.nc(new, "NC_GLOBAL", "history", "NC_CHAR", paste("Original NetCDF file created on
      ",Sys.Date(),sep=""));


      # final operations
      # Sync edited data to disk
      sync.nc(new);
      # Close the netcdf file
      close.nc(new);
    }
  }
}
----------------------------------------------Code ends----------------------------------------------
```

Extreme indices calculation

From the downscaled GCM dataset, we can now calculate the changes in the future precipitation and temperature for our study area. We can also calculate the extreme climatic indices from the downscaled GCM datasets using CDO. For the purpose of the NAP report, we have calculated 11 indices (5 for precipitation and 6 for temperature) using following script.

Note: The definition of future climate extremes used here is based on DHM (2017) and MoFE (2019), and may vary from other literature.

```
---------------------------------------------Code begins---------------------------------------------
## Script to calculate Extreme Climate Indices
rm(list = ls())

library(raster)
library(ncdf4)

input_folder <- "D:\\1.GCM_NAP\\Step3\\"

models <- list.files(input_folder)

output_folder <- "D:\\1.GCM_NAP\\Extreme_analysis\\"

working_folder <- "D:\\1.GCM_NAP\\working_folder\\"
startyear <- 1981
endyear <- 2100
# Precipitation Indices
for (model in models)
{
  dir.create(paste(output_folder,model,sep=""))
  output_folder <- paste(output_folder,model,"\\",sep="")
  for (i in startyear:endyear)
  {

    file <- (paste(input_folder,model,"\\prec_",i,".nc",sep=""))
    command <- paste("cdo timmin ",file," ",working_folder,"min.nc",sep="")
    system(command)
    command <- paste("cdo timmax ",file," ",working_folder,"max.nc",sep="")
    system(command)
    command <- paste("cdo timpctl,95 ",file," ",working_folder,"min.nc ",working_folder,
    "max.nc ",output_folder,"P95_",i,".nc",sep="")
    system(command)
    command <- paste("cdo timpctl,99 ",file," ",working_folder,"min.nc ",working_folder,
    "max.nc ",output_folder,"P99_",i,".nc",sep="")
    system(command)
    command <- paste("cdo eca_rr1 ",file," ",output_folder,"rainydays_",i,".nc",sep = "")
    system(command)
    command <- paste("cdo eca_cdd ",file," ",output_folder,"CDD_",i,".nc",sep = "")
    system(command)
    command <- paste("cdo eca_cwd ",file," ",output_folder,"CWD_",i,".nc",sep = "")
    system(command)
    print(i)
  }
}
```

```
# Temperature Indices
for (model in models)
{
  output_folder <- paste(output_folder,model,"\\",sep="")
  for (i in startyear:endyear)
  {
    file_tmax <- paste(input_folder,model,"\\tmax_",i,".nc",sep="")
    file_tmin <- paste(input_folder,model,"\\tmin_",i,".nc",sep="")
    file_tavg <- paste(input_folder,model,"\\tavg_",i,".nc",sep="")

    #calculate min and max values of each file
    #tmax
    command <- paste("cdo timmin ",file_tmax," ",working_folder,"tmax_min.nc",sep="")
    system(command)
    command <- paste("cdo timmax ",file_tmax," ",working_folder,"tmax_max.nc",sep="")
    system(command)
    #tmin
    command <- paste("cdo timmin ",file_tmin," ",working_folder,"tmin_min.nc",sep="")
    system(command)
    command <- paste("cdo timmax ",file_tmin," ",working_folder,"tmin_max.nc",sep="")
    system(command)
    #tavg
    command <- paste("cdo timmin ",file_tavg," ",working_folder,"tavg_min.nc",sep="")
    system(command)
    command <- paste("cdo timmax ",file_tavg," ",working_folder,"tavg_max.nc",sep="")
    system(command)

    #calculate 10 and 90 percentile of each file
    #tmax
    command <- paste("cdo timpctl,10 ",file_tmax," ",working_folder,"tmax_min.nc ",working_folder,
    "tmax_max.nc ",working_folder,"temp_tmax_10.nc",sep="")
    system(command)

    command <- paste("cdo timpctl,90 ",file_tmax," ",working_folder,"tmax_min.nc ",working_folder,
    "tmax_max.nc ",working_folder,"temp_tmax_90.nc",sep="")
    system(command)

    #tmin
    command <- paste("cdo timpctl,10 ",file_tmin," ",working_folder,"tmin_min.nc ",working_folder,
    "tmin_max.nc ",working_folder,"temp_tmin_10.nc",sep="")
    system(command)

    command <- paste("cdo timpctl,90 ",file_tmin," ",working_folder,"tmin_min.nc ",working_folder,
    "tmin_max.nc ",working_folder,"temp_tmin_90.nc",sep="")
    system(command)

    #cold days calculation
    command <- paste("cdo -add ",working_folder,"temp_tmax_10.nc -sub ",file_tmax," ",file_tmax,"
    ",working_folder,"temp_tmax_10_timesteps.nc",sep = "")
    system(command)
    command <- paste("cdo eca_tg10p ",file_tmax," ",working_folder,"temp_tmax_10_timesteps.nc ",
    output_folder,"Colddays_",i,".nc",sep = "")
    system(command)

    #warm days calculation
    command <- paste("cdo -add ",working_folder,"temp_tmax_90.nc -sub ",file_tmax," ",file_tmax,"
    ",working_folder,"temp_tmax_90_timesteps.nc",sep = "")
    system(command)
    command <- paste("cdo eca_tg90p ",file_tmax," ",working_folder,"temp_tmax_90_timesteps.nc ",
    output_folder,"Warmdays_",i,".nc",sep = "")
    system(command)

    #cold nights calculation
    command <- paste("cdo -add ",working_folder,"temp_tmin_10.nc -sub ",file_tmin," ",file_tmin,"
    ",working_folder,"temp_tmin_10_timesteps.nc",sep = "")
    system(command)
```

```
    command <- paste("cdo eca_tn10p ",file_tmin," ",working_folder,"temp_tmin_10_timesteps.nc ",
    output_folder,"Coldnights_",i,".nc",sep = "")
    system(command)

    #warm nights calculation
    command <- paste("cdo -add ",working_folder,"temp_tmin_90.nc -sub ",file_tmin," ",file_tmin,"
    ",working_folder,"temp_tmin_90_timesteps.nc",sep = "")
    system(command)

    command <- paste("cdo eca_tn90p ",file_tmin," ",working_folder,"temp_tmin_90_timesteps.nc ",
    output_folder,"Warmnights_",i,".nc",sep = "")
    system(command)

    #warm spell calculation
    # infile1: is the base period file (1961-1990) .nc file for which 90th percentile of running 5day
    window is to be computed
    infile1 <- (paste(input_folder,"tasmax_1961-1990.nc",sep=""))
    command <- paste("cdo ydrunmin,5 ",infile1," ",working_folder,"ydrunmin.nc",sep="")
    system(command)
    command <- paste("cdo ydrunmax,5 ",infile1," ",working_folder,"ydrunmax.nc",sep="")
    system(command)
    command <- paste("cdo ydrunpctl,90,5 ",infile1," ",working_folder,"ydrunmin.nc ",working_folder,
    "ydrunmax.nc ",output_folder,"base_pctl90_5day.nc",sep="")
    system(command)
    command <- paste("cdo eca_hwfi,6 ",file_tmax," ",output_folder,"base_pctl90_5day.nc ",
     working_folder,"Warmspell_",year,".nc",sep="")
     system(command)

    #cold spell calculation
    # infile1: is the base period file (1961-1990) .nc file for which 10th percentile of running
    5day window is to be computed
    infile1 <- (paste(input_folder,"tasmin_1961-1990.nc",sep=""))
    command <- paste("cdo ydrunmin,5 ",infile1," ",working_folder,"ydrunmin.nc",sep="")
    system(command)
    command <- paste("cdo ydrunmax,5 ",infile1," ",working_folder,"ydrunmax.nc",sep="")
    system(command)
    command <- paste("cdo ydrunpctl,10,5 ",infile1," ",working_folder,"ydrunmin.nc ",working_folder,
    "ydrunmax.nc ",output_folder,"base_pctl10_5day.nc",sep="")
    system(command)
    command <- paste("cdo eca_cwfi,6 ",file_tmin," ",output_folder,"base_pctl10_5day.nc ",
    working_folder,"Coldspell_",year,".nc",sep="")
  system(command)
    print(i)

  }
  print(paste(model,"done"))
}
-----------------------------------------------Code ends---------------------------------------------
```

References

Bo, J., Terray, L., Habets, F., & Martin, E. (2007). Statistical and dynamical downscaling of the Seine basin climate for hydro-meteorological studies. In *International Journal of Climatology*, 1655 (August), 1643–55. Retrieved from http://doi.org/10.1002/joc.1602

Trzaska, S., & Schnarr, E. (2014). A review of downscaling methods for climate change projections. United States Agency for International Development by Tetra Tech ARD, 1-42. Retrieved from https://www.climatelinks.org/sites/default/files/asset/document/Downscaling_CLEARED.pdf

Déqué, M. (2007). Frequency of precipitation and temperature extremes over France in an anthropogenic scenario: Model results and statistical correction according to observed values. *Global and Planetary Change*, 57(1–2), 16–26. Retrieved from http://doi.org/10.1016/j.gloplacha.2006.11.030

Fowler, H. J., Blenkinsop, S., & Tebaldi, C. (2007). Linking climate change modelling to impacts studies: Recent advances in downscaling techniques for hydrological modelling. *International Journal of Climatology*, (September), 1547–78. Retrieved from http://doi.org/10.1002/joc.1556

Ihaka, R., & Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5:3, 299– 314. Retrieved from http://dx.doi.org/10.1080/10618600.1996.10474713

Immerzeel, W., Pellicciotti, F., & Bierkens, M.F.P. (2013). Rising river flows throughout the twenty-first century in two Himalayan glacierized watersheds. *Nature Geoscience*, 6(9), 742–45. Retrieved from http://doi.org/10.1038/ngeo1896

Lutz, A., & Immerzeel, W. (2015). HI-AWARE reference climate dataset for the Indus, Ganges and Brahmaputra river basins. *FutureWater*, 34. http://lib.icimod.org/record/32311/files/HIAWARE WP2.pdf

Lutz, A., Maat, H., Biemans, H., Shrestha, A., Wester, P., & Immerzeel, W. (2016). Selecting representative climate models for climate change impact studies: An advanced envelope-based selection approach. *International Journal of Climatology*, 36, 1–18. Retrieved from http://doi.org/10.1002/joc.4608

Maraun, D., Wetterhall, F., Chandler, R.E., Kendon, E.J., Widmann, M., Brienen, S., … Thiele-Eich, I. (2010). Precipitation downscaling under climate change: Recent developments to bridge the gap between dynamical models and the end user. *Reviews of Geophysics*, 48(2009RG000314), 1–38. Retrieved from http://doi.org/10.1029/2009RG000314

MoFE. (2019). Climate change scenarios for Nepal. Ministry of Forests and Environment, Nepal. http://www.mofe.gov.np/downloadfile/MOFE_2019_Climate change scenarios for Nepal_NAP_1562647620.pdf

Peterson, T.C. (2005). Climate change indices. *WMO Bulletin*, 54(2), 83–86. Retrieved from http://etccdi.pacificclimate.org/papers/WMO.Bulletin.April.2005.indices.pdf

Piani, C., Haerter, J.O., & Coppola, E. (2010). Statistical bias correction for daily precipitation in regional climate models over Europe. *Theoretical and Applied Climatology*, 187–192. Retrieved from http://doi.org/10.1007/s00704-009-0134-9

Sillmann, J., Kharin, V.V., Zhang, X., Zwiers, F.W., & Bronaugh, D. (2013a). Climate extremes indices in the CMIP5 multimodel ensemble: Part 1. Model evaluation in the present climate. *Journal of Geophysical Research Atmospheres*, 118(4), 1716–33. Retrieved from http://doi.org/10.1002/jgrd.50203

Sillmann, J., Kharin, V.V., Zhang, X., Zwiers, F.W., & Bronaugh, D. (2013b). Climate extremes indices in the CMIP5 multimodel ensemble: Part 2. Future climate projections. *Journal of Geophysical Research Atmospheres*, 118(6), 2473–93. Retrieved from http://doi.org/10.1002/jgrd.50188

Themeßl, M., Gobiet, A., & Heinrich, G. (2011a). Empirical-statistical downscaling and error correction of regional climate models and its impact on the climate change signal. *Climatic Change*, 112(2), 449–68. Retrieved from http://doi.org/10.1007/s10584-011-0224-4

Themeßl, M., Gobiet, A., & Leuprecht, A. (2011b). Empirical-statistical downscaling and error correction of daily precipitation from regional climate models. *International Journal of Climatology*, 31(10), 1530–44. Retrieved from http://doi.org/10.1002/joc.2168

Wilby, R., & Wigley, T. (1997). Downscaling general circulation model output: A review of methods and limitations. **Progress in Physical Geography**, 21(4), 530–48. Retrieved from http://doi.org/10.1177/030913339702100403

About ICIMOD

The International Centre for Integrated Mountain Development (ICIMOD), is a regional knowledge development and learning centre serving the eight regional member countries of the Hindu Kush Himalaya – Afghanistan, Bangladesh, Bhutan, China, India, Myanmar, Nepal, and Pakistan – and based in Kathmandu, Nepal. Globalisation and climate change have an increasing influence on the stability of fragile mountain ecosystems and the livelihoods of mountain people. ICIMOD aims to assist mountain people to understand these changes, adapt to them, and make the most of new opportunities, while addressing upstream-downstream issues. We support regional transboundary programmes through partnership with regional partner institutions, facilitate the exchange of experience, and serve as a regional knowledge hub. We strengthen networking among regional and global centres of excellence. Overall, we are working to develop an economically and environmentally sound mountain ecosystem to improve the living standards of mountain populations and to sustain vital ecosystem services for the billions of people living downstream – now, and for the future.

REGIONAL MEMBER COUNTRIES

AFGHANISTAN BANGLADESH BHUTAN CHINA

INDIA MYANMAR NEPAL PAKISTAN

SUPPORTED BY

ICIMOD gratefully acknowledges the support of its core donors: the Governments of Afghanistan, Australia, Austria, Bangladesh, Bhutan, China, India, Myanmar, Nepal, Norway, Pakistan, Sweden, and Switzerland.

International Centre for Integrated Mountain Development
GPO Box 3226, Kathmandu, Nepal
T +977 1 5275222 | **E** info@icimod.org | www.icimod.org

ISBN 978 92 9115 6887 (online)