

# 기후자료 처리 인터페이스 서비스

(OpenWPS: Open Web Processing Service for the  
Climate Science Community)

APEC 기후센터

2020. 10. 07



# 목차

1. 개요
2. 시스템 구성
3. 자료구성
4. 주요기능
5. 향후,개선계획
6. 활용



# 기후자료 처리 인터페이스 서비스(OpenWPS)\_정의

- OpenWPS란 ?

- 기후자료 처리기술이 부족한 기후과학 커뮤니티(Climature Science Community)에 기후자료를 처리하는데 중요한 기반 기술을 제공하는 서비스
- **Open Web Processing Service** for the Climate Science Community  
(기후자료 처리 인터페이스 서비스)

- 필요성

- 기후자료는 공간정보를 포함하고 있어 일반 사용자들이 사용하기 어려움
- 기후정보 시각화/가공에 많은 시간과 자료처리 전문기술이 필요함

- 목적

- 공간정보 분야 국제표준 개방형 공간정보 컨소시엄의 웹 프로세싱 기술 제공
- 인터넷 환경이 열악한 환경에서도 이용 가능한 서비스 제공
- 운영체제, 개발언어에 상관없이 국제표준 기반 인터페이스 서비스 제공

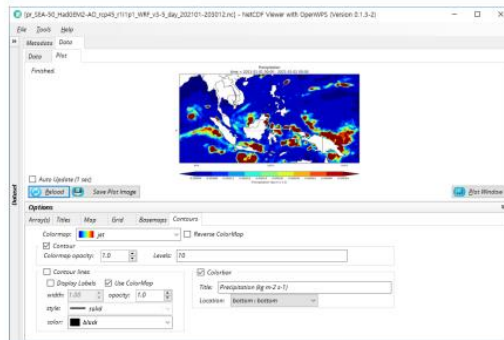
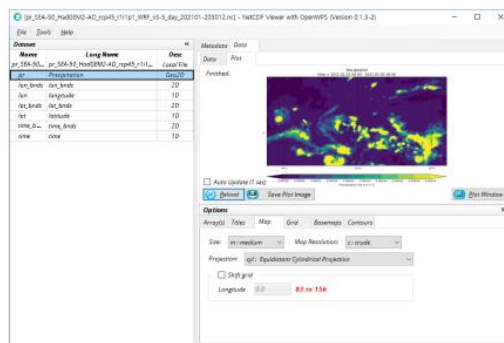
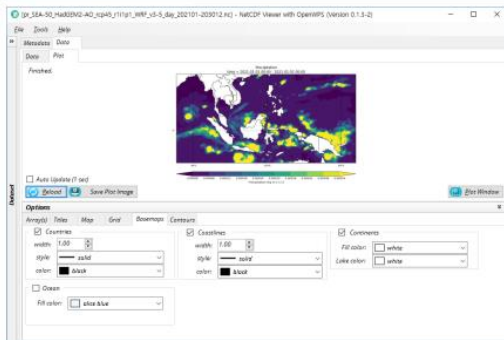
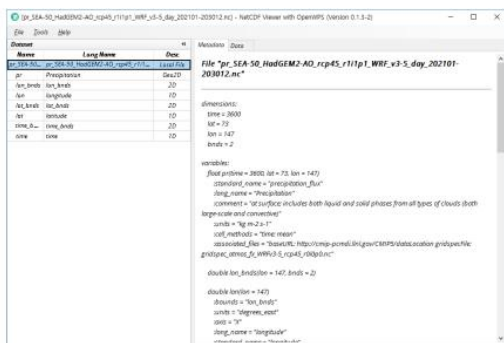
- 기후자료

- ADSS, CLIPs에서 제공하는 기후자료를 활용

# 기후자료 처리 인터페이스 서비스(OpenWPS)\_서비스

## • 기후자료 시각화 서비스

– 기후자료 시각화, 공간정보 포맷 자료 변환 제공

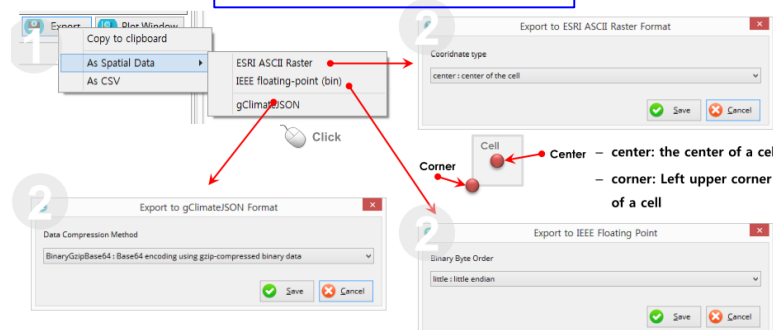


NetCDF 파일

공간정보 자료  
파일포맷 변환

ESRI ASCII Raster  
IEEE floating-point(bin)  
gClimateJSON

• Attribute View > Data Tab



# 기후자료 처리 인터페이스 서비스(OpenWPS)\_서비스

## • 마스킹 기법을 이용한 기후자료 추출 서비스

### - 주요기능

- 사용자 작업관리, 자료검색, 자료추출, 자료가공, 시각화 제공

OpenWPS:CP MaskWithISO3166

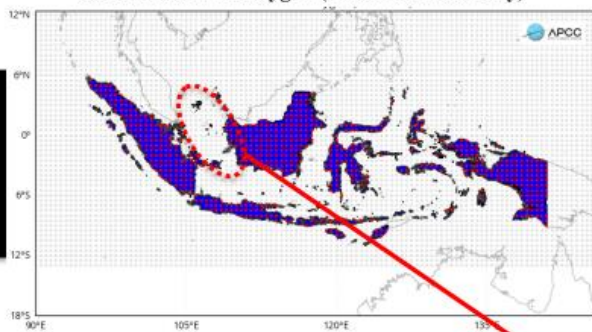
(ISO-3166 국가코드 기반 기후자료

마스킹) 인터페이스 적용

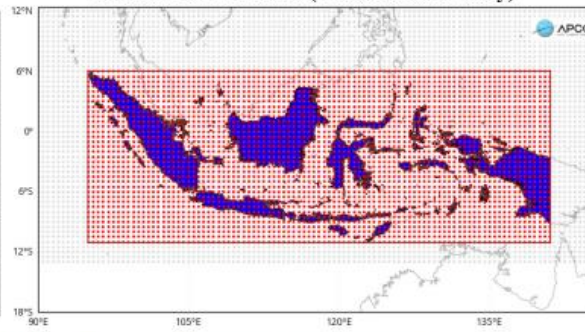
GIS 소프트웨어(GDAL)

Rectangle 방식 추출

Case 1: Point in Polygon (Indonesia boundary)



Case 2: Point in MBR (Indonesia boundary)

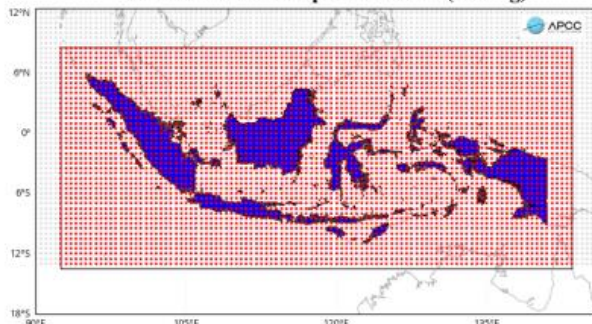


Rectangle 방식 추출

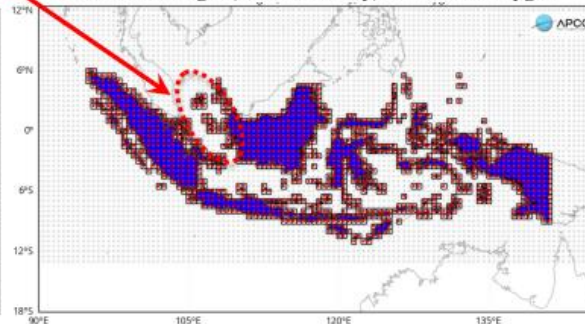
작은 섬까지  
정확하게 선택됨

제한 알고리즘

Case 3: Point in Expanded MBR (2.5 deg)



Case 4: Rectangle (Cell boundary) intersects Polygon



```
openwps-tb
├── src/main/java
│   ├── (default package)
│   ├── org.apcc21.openwps.tb.climateMask
│   │   ├── OTBNMaskWithCF.java
│   │   └── OTBNMaskWithCF.java
│   ├── org.apcc21.openwps.tb.configuration
│   │   ├── OTBNWPSConfig.java
│   │   ├── OTBNWPSConfig.java
│   │   └── OTBNWPSConfig.java
│   ├── org.apcc21.openwps.tb.gClimateJSON
│   │   ├── OCDataMethod.java
│   │   ├── OCDataMethod.java
│   │   └── OCDataMethod.java
│   ├── org.apcc21.openwps.tb.NetCDF.CF.Variable
│   │   ├── OTBNCCFVarData1D.java
│   │   ├── OTBNCCFVarDimIndex.java
│   │   ├── org.apcc21.openwps.tb.NetCDF.Mask
│   │   │   ├── OTBNMaskWriter.java
│   │   │   ├── OTBNMaskWriter.java
│   │   │   └── OTBNMaskWriter.java
│   │   ├── org.apcc21.openwps.tb.NetCDF.netcdftime
│   │   │   ├── datetime.java
│   │   │   ├── DateTimeDayOfMonthExtensions.java
│   │   │   ├── netcdftime.java
│   │   │   └── utime.java
│   ├── org.apcc21.openwps.tb.NetCDF.python
│   │   ├── Python_Math.java
│   │   ├── Python_String.java
│   │   └── Python_Time.java
│   ├── org.apcc21.openwps.tb.NetCDF.Read
│   │   ├── OTBNReader.java
│   │   ├── OTBNReader4CF.java
│   │   ├── org.apcc21.openwps.tb.OpenWPS
│   │   │   ├── OPMaskingOperator.java
│   │   │   └── OPMaskingOperator.java
│   ├── org.apcc21.openwps.tb.sample
│   │   ├── S_MaskWithCF.java
│   │   └── S_MaskWithCF.java
│   ├── org.apcc21.openwps.tb.util
│   │   ├── ArrayUtil.java
│   │   ├── FileUtil.java
│   │   ├── GzipUtil.java
│   │   ├── PrintUtil.java
│   │   └── StringUtil.java
```

프로그램 소스코드 구조

CORDEX-SEA25  
(by Dr. Hongwei Yang, APCC)

# 기후자료처리 인터페이스 서비스(OpenWPS)\_SW 활용방법

- OpenWPS 웹사이트 접속: <http://openwps.apcc21.org>
  - 웹 사이트를 통해 프로그램 및 사용매뉴얼 제공

OpenWPS

WHAT WPS SERVICES TUTORIAL APPLICATION

## What is OpenWPS ?

OpenWPS is a climate-data-specific service using OGC (Open geospatial consortium) WPS (web processing service) that is an international standard in spatial information field. Users can use OpenWPS using three operations that are GetCapabilities, DescribeProcess, and Execution on any environment (e.g. graphical user interface program, server system, web service, etc.) more easily and conveniently.

## WPS Services

You points your WPS client to <http://openwps.apcc21.org/wps>

OpenWPS [GetCapabilities](#) OpenWPS [DescribeProcess](#)

**OpenWPS:CV\_VisualizeNonSeries**  
returns map-based plot image based on the inputs [DOC](#)

Input Parameter	Description	Data Type
inputData	Climate data for visualization in gClimateJSON format	application/json
plotOption	Plot options for visualization	application/json

**OpenWPS:CP\_MaskWithCF**  
returns mask data generated base on the inputs [DOC](#)

OpenWPS 프로그램

매뉴얼(한,영)

개인 컴퓨터에 설치하여 사용

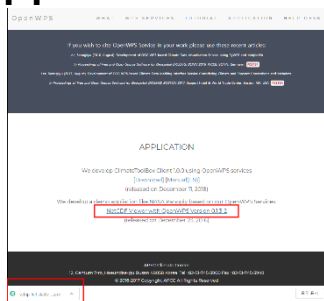


# 실습(프로그램 실행)

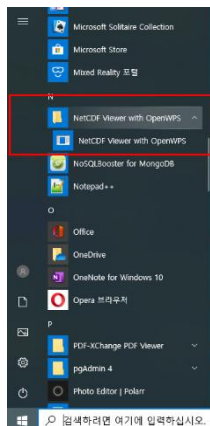
1. 웹 브라우저에서 OpenWPS 웹 사이트(openwps.apcc21.org) 접속



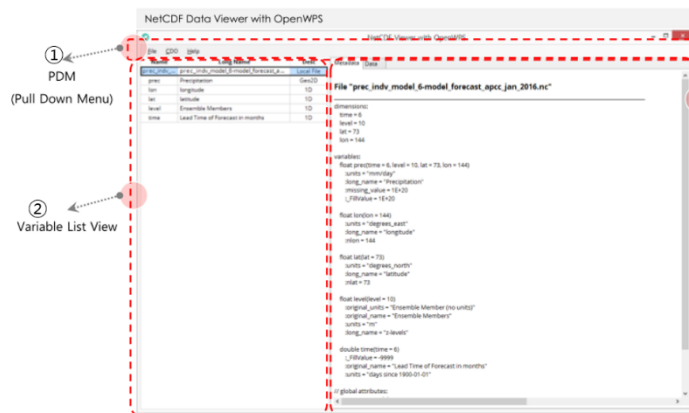
2. Application 프로그램 다운로드



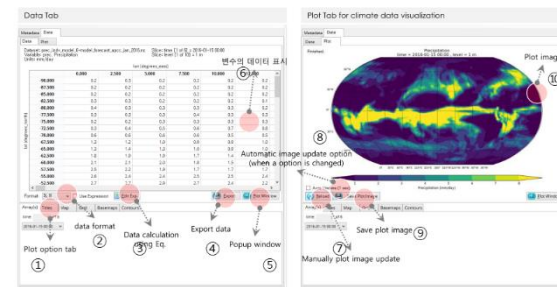
3. 실행파일 설치



4. 사용자 화면에서 파일 불러오기

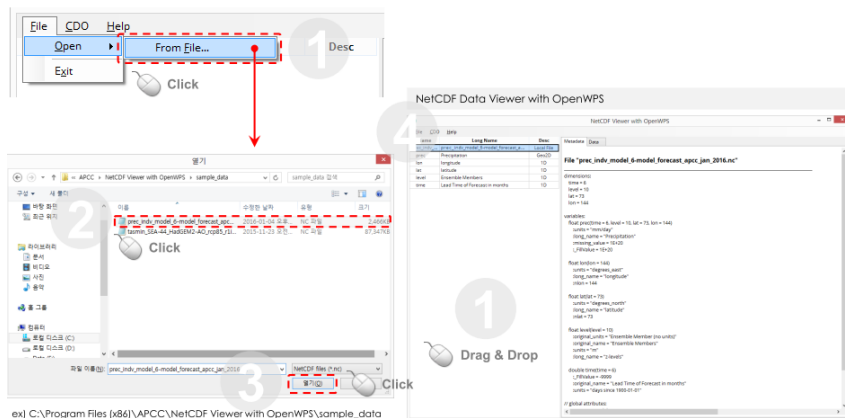


5. 자료처리 시각화 화면

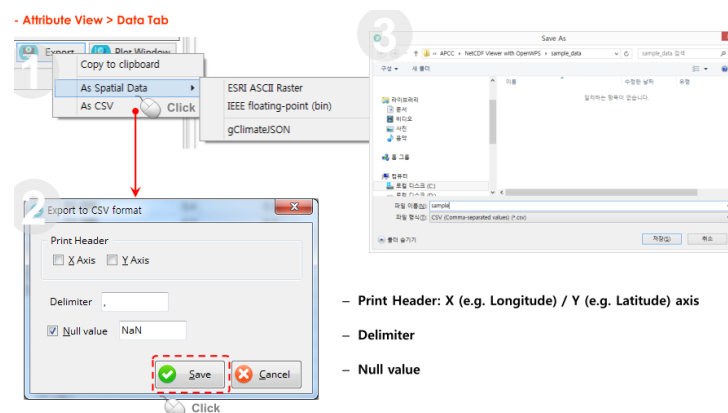


# 실습(파일변환)

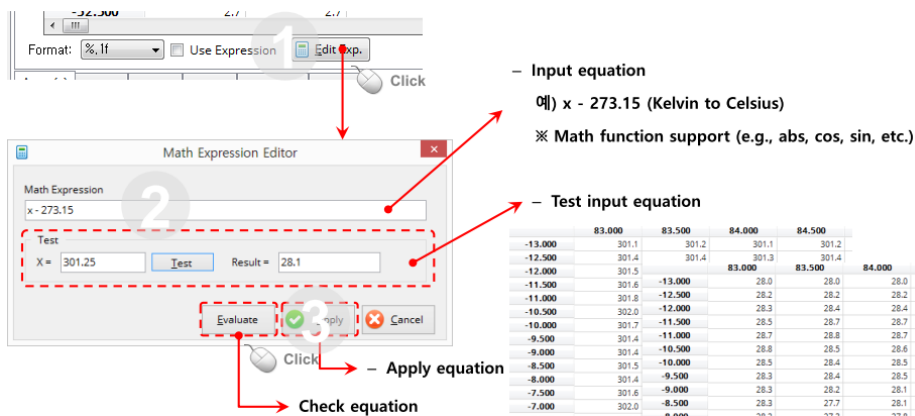
## 1. 파일 불러오기



## 3. 텍스트 자료파일 포맷 변환



## 2. 수학적 계산하기



칼빈(kelvin)을 섭씨(Celsius)로 변환할 수 있음.



# 실습(Masking)

<http://openwps.apcc21.org/tutorial/tutorial3.html>

← 파이썬 실습코드

## gClimateMaskJson 파일 생성

```
In [1]: import tutorial_util as util

In [2]: from netCDF4 import Dataset
import numpy as np

In [3]: ncFile = 'data/pr_SEA-44_040002-02_rcp85_r11p1_00V_v3-5_day_200101-210012.nc'

In [4]: ds = Dataset(ncFile, 'r') # Open netcdf file

In [5]: lon = ds.variables['lon'][:] # Get data of lon variable in the nc file

In [6]: lat = ds.variables['lat'][:] # Get data of lat variable in the nc file

In [7]: lon_bounds = ds.variables['lon_bnds'][:] # Get data of lon_bnds variable in the nc file

In [8]: lat_bounds = ds.variables['lat_bnds'][:] # Get data of lat_bnds variable in the nc file

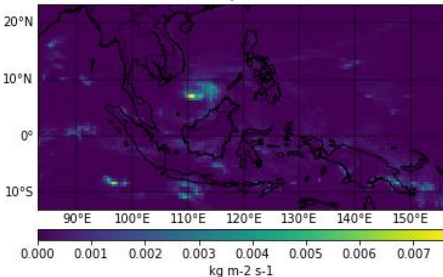
In [9]: # For plotting
pr = ds.variables['pr'][:,0,:] # time index = 1
pr_units = ds.variables['pr'].units

In [10]: # Generating gClimateMaskJSON data using variables in the nc file
inputClimateMask = util.new_gClimateMaskJSON(lat, lon, lat_bounds, lon_bounds)

In [11]: print (inputClimateMask)
```



Precipitation



## Shape 파일 읽기

Reading shape file

```
In [13]: lat_min = -12.0
lat_max = 30.0

lon_min = 83.0
lon_max = 156.0

In [14]: # Thailand (tha) # tha is ISO-3166-1 alpha-3 code
shape_file = 'data/shape/tha_simplified'

In [15]: # Lao (lao) # lao is ISO-3166-1 alpha-3 code
shape_file = 'data/shape/lao_simplified'

In [16]: # Plotting shape data
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt

map = Basemap(projection='merc', resolution='l',
              llcrnrlat=lat_min, urcnrlat=lat_max, \
              llcrnrlon=lon_min, urcnrlon=lon_max, \
              lat_ts=0)

#map.drawmapboundary(fill_color='aqua')
#map.fillcontinents(color='#ddaa66', lake_color='aqua')
#map.drawcoastlines()

map.readshapefile(shape_file, 'country')

plt.show()
```



## Shape값을 GeoJSON 변환

Shapefile to GeoJSON

```
In [17]: inputRegion = util.new_shapefileGeoJSON(shape_file)

In [18]: print (inputRegion)

{
  "type": "FeatureCollection",
  "features": [
    {
      "properties": {
        "NAME": "TH",
        "YINHT_120": "120",
        "YINHT_180": "180"
      },
      "type": "Feature",
      "geometry": {
        "coordinates": [
          [
            101.77647333333333,
            12.449273444444444,
            101.87298333333333,
            22.303612777777778,
            101.12039444444444,
            22.453232555555556,
            100.43323255555556,
            22.500000000000004,
            100.27000000000000,
            21.859393888888889,
            100.27000000000000,
            21.859393888888889
          ]
        ]
      }
    }
  ]
}
```

## OpeWPS 활용 마스킹 자료 추출

Making mask data with APCC OpenWPS service

```
In [19]: from netlib.wps import WebProcessingService

In [20]: from netlib.wps import WebProcessingService, monitorExecution

In [21]: import tutorial_util as util

In [22]: service_host = 'http://openwps.apcc21.org/ops/ops.html'

In [23]: wps_process = 'OpenWPS-CF_Precipitation' # WPS process name

In [24]: wps = WebProcessingService(service_host, verbose=False, skip_req=True)

In [25]: inputMaskingOperator = 'RectInterp' # 'PctInterp' # 'PctPoly' # 'RectInterp'
inputDistance = '7.5'

In [26]: inputs = [
    ('inputClimateMaskData', inputClimateMask),
    ('inputRegion', inputRegion),
    ('inputMaskingOperator', inputMaskingOperator),
    ('inputDistance', inputDistance)
]

In [27]: outputs = [
    ('maskData', 'output'),
    ('reference', False)
]

In [28]: execution = wps.execute(wps_process, inputs, output=outputs)

In [29]: monitorExecution(execution)

In [30]: mask_data = execution.processOutputs[0]
print(mask_data.data)

[{"maskResult": [{"version": "0.1", "dimension": [73, 147], "method": "Bilinear", "mask": "RectInterp"}]}]

In [31]: print (mask_data.data)

{"maskResult": [{"version": "0.1", "dimension": [73, 147], "method": "Bilinear", "mask": "RectInterp"}]}
```

```
masked_array(data =
[[[...]]],
mask =
[[[...]]],
fill_value = 1e+20)
```

## 마스킹 자료 적용한 자료 표출

```
In [44]: # Plot Data
masked_pr = np.ma.masked_where(mask_data_array==False, pr)
cs = m.pcolormap(xl,y1,np.squeeze(masked_pr))

#clevis = [0,1,2,5,5,7,5,10,15,20,30,40,50,70,100,150,200,250,300,400,500,600,750]
#cs = m.contourf(xl, yl, pr, clevis, cmap=cmap, cspan=3)

# Add Grid Lines
m.drawparallels(np.arange(-80., 81., 10.), labels=[1,0,0,0], fontsize=10)
m.drawmeridians(np.arange(-180., 181., 10.), labels=[0,0,0,1], fontsize=10)

# Add Coastlines, States, and Country Boundaries
m.drawcoastlines()
m.drawstates()
m.drawcountries()

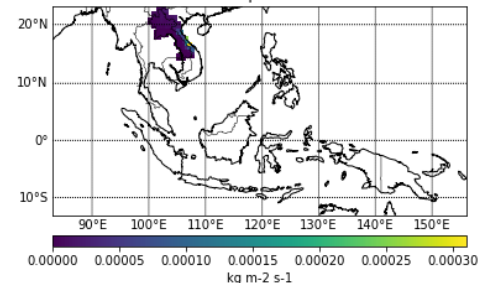
# Add Colorbar
cbar = m.colorbar(cs, location='bottom', pad='10%')
cbar.set_label(pr_units)

# Add Title
plt.title('Precipitation')

plt.show()
```



Precipitation



- 
- NetCDF
- CF-Meta Convention



# 웹 기반 자료합성(composite) 서비스 제공(11월 예정)

APCC

### FORECAST Multi Model Ensemble(SCM)

Variable: Precipitation Forecast length: 6Month MME

Month: ☐ 01 ☐ 02 ☐ 03 ☐ 04 ☐ 05 ☐ 06 ☐ 07 ☐ 08 ☐ 09 ☐ 10 ☐ 11 ☐ 12

Years: ☐ 2015 ☐ 2016 ☐ 2017 ☐ 2018 ☐ 2019 ☐ 2020

Lead time: ☐ 01 ☐ 02 ☐ 03 ☐ 04 ☐ 05 ☐ 06

[Add](#) [Reset](#) [Uncheck All](#)

Requirements

And	month	year	lead time
<input type="checkbox"/>	05	2018	01

[Composite/Forecast](#)

Monthly Precipitation (2015/01)

Period: 201501-201503

[NetCDF \(.nc\)](#) [ASCII \(.txt\)](#) [Image \(.png\)](#)

### OBSERVATION Monthly data

Variable: Precipitation ☒ mean ☐ Anomaly 1979 - 1979

Month: ☐ 01 ☐ 02 ☐ 03 ☐ 04 ☐ 05 ☐ 06 ☐ 07 ☐ 08 ☐ 09 ☐ 10 ☐ 11 ☐ 12

Years: ☐ 1979 ☐ 1980 ☐ 1981 ☐ 1982 ☐ 1983 ☐ 1984 ☐ 1985 ☐ 1986 ☐ 1987 ☐ 1988 ☐ 1989 ☐ 1990 ☐ 1991 ☐ 1992 ☐ 1993 ☐ 1994 ☐ 1995 ☐ 1996 ☐ 1997 ☐ 1998 ☐ 1999 ☐ 2000 ☐ 2001 ☐ 2002 ☐ 2003 ☐ 2004 ☐ 2005 ☐ 2006 ☐ 2007 ☐ 2008 ☐ 2009 ☐ 2010 ☐ 2011 ☐ 2012 ☐ 2013 ☐ 2014 ☐ 2015 ☐ 2016 ☐ 2017 ☐ 2018 ☐ 2019 ☐ 2020

[Add](#) [Reset](#) [Uncheck All](#)

Requirements

And	Month	year
<input type="checkbox"/>	01.03.05	2017,2018,2019

[Composite/Observation](#)

Monthly Sea Level Pressure (2015/01)

Period: 201501-201503

[NetCDF \(.nc\)](#) [ASCII \(.txt\)](#) [Image \(.png\)](#)

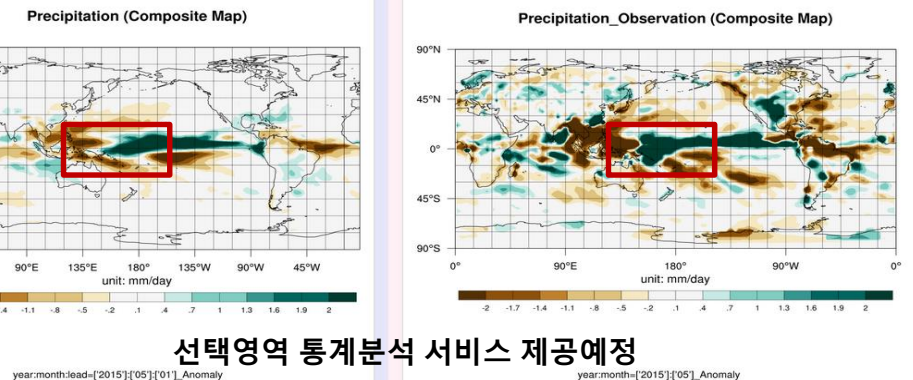
About CLIK Disclaimer Privacy Cookies Copyright © 2019 APEC Climate Center

<http://cliks.apcc21.org>

계절예측자료+Anomaly 합성

관측자료+Mean, Anomaly 계산 + 합성

마스킹 적용 자료추출 + 통계정보('21)



선택영역 통계분석 서비스 제공예정

파일 내려받기(NetCDF,Xls)

Download Data for 201501-201503 (2015/01-2015/03) (Unit: mm/day)

Year	Month	Lead	Value
2015	01	01	0.1171
2015	01	02	0.1171
2015	01	03	0.1171
2015	01	04	0.1171
2015	01	05	0.1171
2015	01	06	0.1171
2015	01	07	0.1171
2015	01	08	0.1171
2015	01	09	0.1171
2015	01	10	0.1171
2015	01	11	0.1171
2015	01	12	0.1171
2015	02	01	0.1171
2015	02	02	0.1171
2015	02	03	0.1171
2015	02	04	0.1171
2015	02	05	0.1171
2015	02	06	0.1171
2015	02	07	0.1171
2015	02	08	0.1171
2015	02	09	0.1171
2015	02	10	0.1171
2015	02	11	0.1171
2015	02	12	0.1171
2015	03	01	0.1171
2015	03	02	0.1171
2015	03	03	0.1171
2015	03	04	0.1171
2015	03	05	0.1171
2015	03	06	0.1171
2015	03	07	0.1171
2015	03	08	0.1171
2015	03	09	0.1171
2015	03	10	0.1171
2015	03	11	0.1171
2015	03	12	0.1171
2015	04	01	0.1171
2015	04	02	0.1171
2015	04	03	0.1171
2015	04	04	0.1171
2015	04	05	0.1171
2015	04	06	0.1171
2015	04	07	0.1171
2015	04	08	0.1171
2015	04	09	0.1171
2015	04	10	0.1171
2015	04	11	0.1171
2015	04	12	0.1171
2015	05	01	0.1171
2015	05	02	0.1171
2015	05	03	0.1171
2015	05	04	0.1171
2015	05	05	0.1171
2015	05	06	0.1171
2015	05	07	0.1171
2015	05	08	0.1171
2015	05	09	0.1171
2015	05	10	0.1171
2015	05	11	0.1171
2015	05	12	0.1171
2015	06	01	0.1171
2015	06	02	0.1171
2015	06	03	0.1171
2015	06	04	0.1171
2015	06	05	0.1171
2015	06	06	0.1171
2015	06	07	0.1171
2015	06	08	0.1171
2015	06	09	0.1171
2015	06	10	0.1171
2015	06	11	0.1171
2015	06	12	0.1171
2015	07	01	0.1171
2015	07	02	0.1171
2015	07	03	0.1171
2015	07	04	0.1171
2015	07	05	0.1171
2015	07	06	0.1171
2015	07	07	0.1171
2015	07	08	0.1171
2015	07	09	0.1171
2015	07	10	0.1171
2015	07	11	0.1171
2015	07	12	0.1171
2015	08	01	0.1171
2015	08	02	0.1171
2015	08	03	0.1171
2015	08	04	0.1171
2015	08	05	0.1171
2015	08	06	0.1171
2015	08	07	0.1171
2015	08	08	0.1171
2015	08	09	0.1171
2015	08	10	0.1171
2015	08	11	0.1171
2015	08	12	0.1171
2015	09	01	0.1171
2015	09	02	0.1171
2015	09	03	0.1171
2015	09	04	0.1171
2015	09	05	0.1171
2015	09	06	0.1171
2015	09	07	0.1171
2015	09	08	0.1171
2015	09	09	0.1171
2015	09	10	0.1171
2015	09	11	0.1171
2015	09	12	0.1171
2015	10	01	0.1171
2015	10	02	0.1171
2015	10	03	0.1171
2015	10	04	0.1171
2015	10	05	0.1171
2015	10	06	0.1171
2015	10	07	0.1171
2015	10	08	0.1171
2015	10	09	0.1171
2015	10	10	0.1171
2015	10	11	0.1171
2015	10	12	0.1171
2015	11	01	0.1171
2015	11	02	0.1171
2015	11	03	0.1171
2015	11	04	0.1171
2015	11	05	0.1171
2015	11	06	0.1171
2015	11	07	0.1171
2015	11	08	0.1171
2015	11	09	0.1171
2015	11	10	0.1171
2015	11	11	0.1171
2015	11	12	0.1171
2015	12	01	0.1171
2015	12	02	0.1171
2015	12	03	0.1171
2015	12	04	0.1171
2015	12	05	0.1171
2015	12	06	0.1171
2015	12	07	0.1171
2015	12	08	0.1171
2015	12	09	0.1171
2015	12	10	0.1171
2015	12	11	0.1171
2015	12	12	0.1171

# 고맙습니다.

## - APEC기후센터 예측운영과

선임연구원 한정민  
goal@apcc21.org



연구원 정임국  
igjung@apcc21.org

