Supplementary Information

for *Dependence of Relative Humidity on the Given Distribution of Relative Humidity*

S. BOURDIN, L. KLUFT and B. STEVENS

1 Reanalyses' Relative Humidity Trend Analysis

This document aims at describing more precisely the trend analysis on reanalysis data. It is available as a Jupyter Notebook at https://doi.org/10.5281/zenodo.4423267, along with the necessary data.

Two datasets have been used for this analysis : ERA5 and JRA-55. ERA5 is available on the Climate Data Store (https://cds.climate.copernicus.eu/cdsapp#!/dataset/reanalysis-era5-pressure-levels-monthly-means?tab=overview) and was in our case retrieved from a miror on the IPSL servers because the amount of data necessary was very long to obtain through the CDS API. JRA-55 was retrieve from their FTP server, now unavailable. Other means to access the data are describe on their website : https://jra.kishou.go.jp/JRA-55/index_en.html.

1.1 Data pre-processing

1.1.1 ERA5

ERA5 relative humidity data was available in monthly files containing 6-hourly time step. For each monthly files, using nco, we (1) averaged the data over each month (`ncra $mthfile r_${date}.nc`), (2) extracted the tropical zone between +/- 30° (`ncks -v r -d latitude,-30.0,+30.0 r_${date}.nc r_$[date}.nc`). Then concatenated all the data in one file (`ncrcat r_*.nc r_ERA.nc`). We also retrieve the corresponding land-sea mask (`lsm_ERA.nc`).

A python script was apply to retrieve the mean tropical profile over oceans for each month (Note : We did not apply a weighted average assuming grid cell area differences in this region was negligible.) :

```
from dynamicopy import var_load
import numpy as np
import pickle as pkl
import os

# Load data
f = 'r_ERA.nc'
H = var_load('r', f)
P = var_load('level', f)

f_lsm = 'lsm_ERA.nc'
```

```
lsm = var_load('lsm', f_lsm)
mask = lsm[0] < 0.1
H_masked = H * mask
H_masked[H_masked == 0.0] = np.nan

H_tropical = np.nanmean(np.nanmean(H_masked, -1), -1)

with open('H_tropical.pkl', 'wb') as handle:
    pkl.dump(H_tropical, handle)
```

1.1.2 JRA-55

Monthly JRA-55 data was available on the FTP server, but in GRIB format. It was copied to NetCDF using `cdo -f nc copy $file ${file}.nc`, then all the file were concatenated before cutting the -30°/+30° latitude zone, and the same script was applied to obtain the tropical mean profile.

1.2 Trend computation

The trends are computed using a linear regression, and dismissing the hypothesis that the trend might be zero.

[1]:
```
import pickle as pkl
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
#Load data
##ERA5
with open("ERA/P_ERA.pkl",'rb' ) as handle :
    P_ERA = pkl.load(handle)
with open("ERA/H_tropical.pkl", "rb") as handle:
    H_trop_ERA = pkl.load(handle)
t_ERA = np.arange(len(H_trop_ERA))/12 + 1979 # Time axis in years
### Cut level above 100hPa
H_trop_ERA = H_trop_ERA[:,P_ERA >= 100]
P_ERA = P_ERA[P_ERA >= 100]
```

[3]:
```
## JRA-55
with open("JRA/P_JRA.pkl",'rb' ) as handle :
    P_JRA = pkl.load(handle)
with open("JRA/H_tropical.pkl", "rb") as handle:
    H_trop_JRA = pkl.load(handle)
t_JRA = np.arange(len(H_trop_JRA))/12 + 1979 # Time axis in years
```

To compute the trend, we use `scipy.stats.linregress` function, whose output is defined as follows in the documentation :

```
    Returns
    -------
    slope : float
        Slope of the regression line.
    intercept : float
        Intercept of the regression line.
    rvalue : float
        Correlation coefficient.
    pvalue : float
        Two-sided p-value for a hypothesis test whose null hypothesis is
        that the slope is zero, using Wald Test with t-distribution of
        the test statistic.
    stderr : float
        Standard error of the estimated gradient.
```

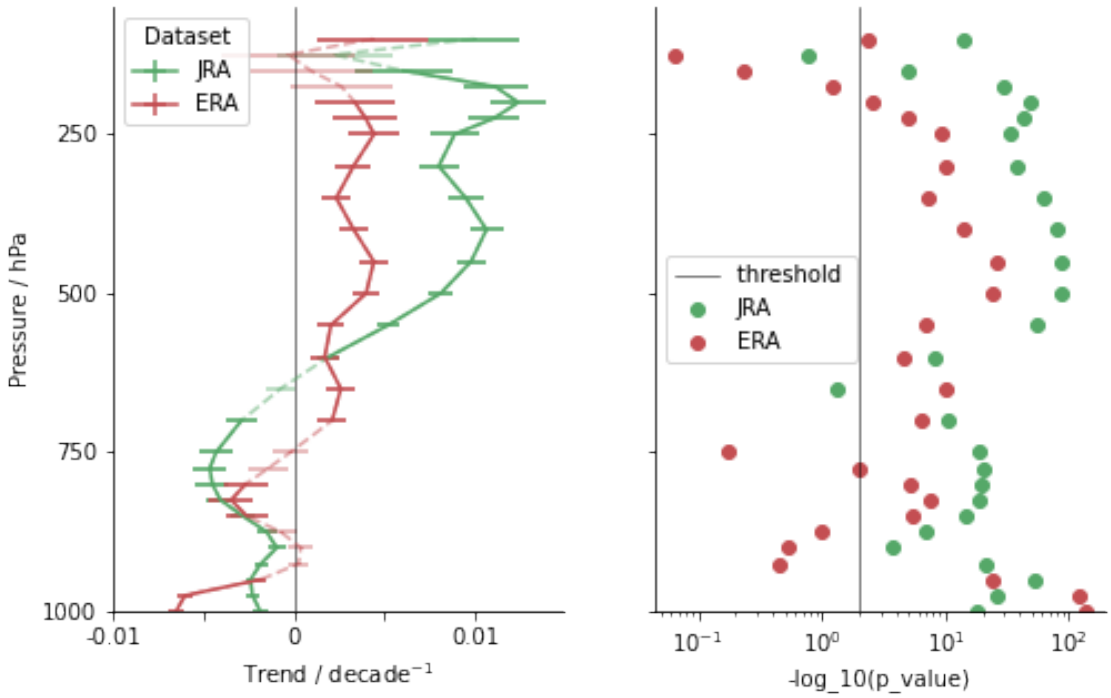
In particular, the "error" or uncertainty refers to the `stderr` which is the standard error for the gradient estimation or 67% confidence interval, and the p-value is computed for a null hypothesis where the slope is zero. We use 0.01 as a discriminating threshold for p-values, but p-values are displayed for you to appreciate.

[4]:
```
# Compute trend
##Function
def compute_linear_trend(time, H) :
    trends, intercepts, p_values, stderrs = [], [], [], []
    for p in range(np.shape(H)[1]) :
        slope, intercept, r, p, stderr = stats.linregress(time, H[:,p])
        trends.append(slope)
        intercepts.append(intercept)
        p_values.append(p)
        stderrs.append(stderr)
    return np.array([trends, intercepts, p_values, stderrs])
```

[5]:
```
## ERA
trends_ERA, intercepts_ERA, pvals_ERA, errs_ERA = compute_linear_trend(t_ERA,␣
 ↪H_trop_ERA)
mtrends_ERA = np.ma.masked_array(trends_ERA, pvals_ERA > 0.01)
```

[6]:
```
## JRA
trends_JRA, intercepts_JRA, pvals_JRA, errs_JRA = compute_linear_trend(t_JRA,␣
 ↪H_trop_JRA)
mtrends_JRA = np.ma.masked_array(trends_JRA, pvals_JRA > 0.01)
```

[7]:
```
[Plotting commands]
```



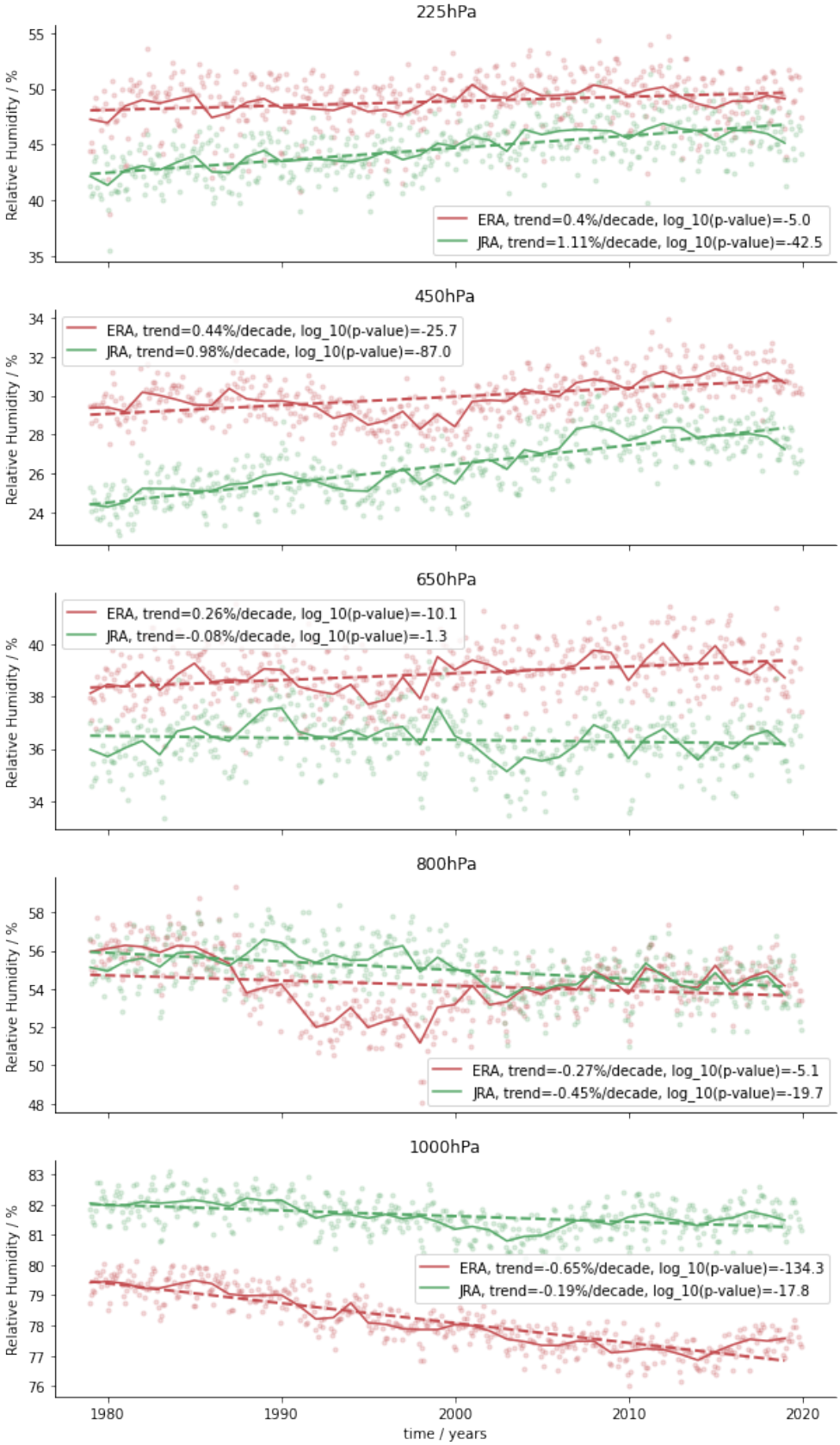
The figure above summarizes our trend analysis : On the left panel is the trend in both reanalysis products, at each levels, with dashed line and transparent points where the p-value > 0.01 does not allow us to conclude that the trend is not zero. Errorbars show 95% confidence interval ($2 \times \sigma$). On the right panel is displayed the logarithm of the p-value for each trend computation, as compared to the threshold.

1.3 Time series

Here we display a few time series to illustrate the trend. Dots indicate the monthly values, solid line the yearly average of these values, and dashed line the linear trend corresponding to what is above.

[8]:
```
def yearly_avg(H) :
    return np.array([np.mean(H[yr*12:(yr+1)*12],0) for yr in range(int(len(H)/
 ↪12))])
H_ERA_yr = yearly_avg(H_trop_ERA)
H_JRA_yr = yearly_avg(H_trop_JRA)
yrs = np.arange(1979, 2020)
```

[9]:
```
[Plotting commands]
```


225hPa
Relative Humidity / %
ERA, trend=0.4%/decade, log_10(p-value)=-5.0
JRA, trend=1.11%/decade, log_10(p-value)=-42.5
450hPa
Relative Humidity / %
ERA, trend=0.44%/decade, log_10(p-value)=-25.7
JRA, trend=0.98%/decade, log_10(p-value)=-87.0
650hPa
Relative Humidity / %
ERA, trend=0.26%/decade, log_10(p-value)=-10.1
JRA, trend=-0.08%/decade, log_10(p-value)=-1.3
800hPa
Relative Humidity / %
ERA, trend=-0.27%/decade, log_10(p-value)=-5.1
JRA, trend=-0.45%/decade, log_10(p-value)=-19.7
1000hPa
Relative Humidity / %
ERA, trend=-0.65%/decade, log_10(p-value)=-134.3
JRA, trend=-0.19%/decade, log_10(p-value)=-17.8
1980
1990
2000
2010
2020
time / years

1.4 Decade differences

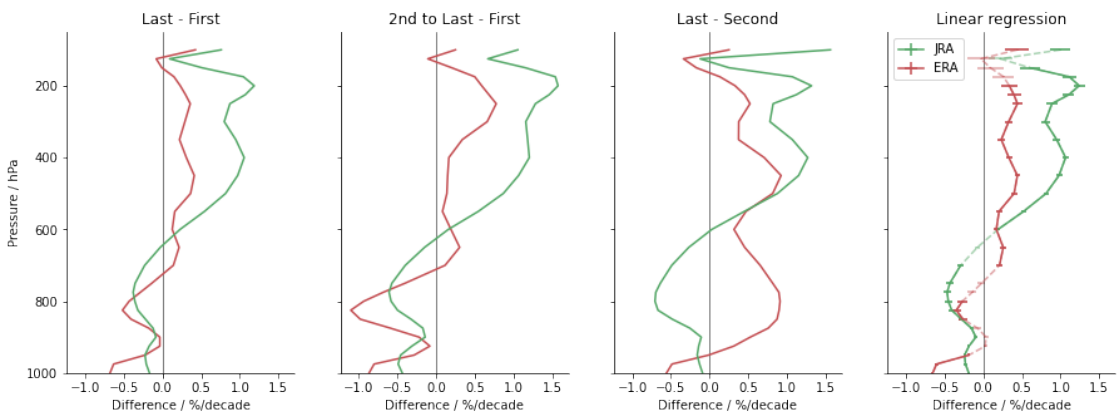
As an other way to check the robustness of the trend we observe here, we computed the differences for both datasets at each levels * between the last (2010-2019) and the first (1979-1988) decades * between the second-to-last (2000-2009) and the first decades. * between the last and the second (1989-1999) decade.

[10]:
```
[Plotting commands]

ax[0].plot((np.mean(H_trop_ERA[-120:],0) - np.mean(H_trop_ERA[:120],0))/3,␣
 ↪P_ERA, c= sns.color_palette("deep")[3])
ax[0].plot((np.mean(H_trop_JRA[-120:],0) - np.mean(H_trop_JRA[:120],0))/3,␣
 ↪P_JRA, c= sns.color_palette("deep")[2])
ax[1].plot((np.mean(H_trop_ERA[-240:-120],0) - np.mean(H_trop_ERA[:120],0))/2,␣
 ↪P_ERA, c= sns.color_palette("deep")[3])
ax[1].plot((np.mean(H_trop_JRA[-240:-120],0) - np.mean(H_trop_JRA[:120],0))/2,␣
 ↪P_JRA, c= sns.color_palette("deep")[2])
ax[2].plot((np.mean(H_trop_ERA[-120:],0) - np.mean(H_trop_ERA[120:240],0))/2,␣
 ↪P_ERA, c= sns.color_palette("deep")[3])
ax[2].plot((np.mean(H_trop_JRA[-120:],0) - np.mean(H_trop_JRA[120:240],0))/2,␣
 ↪P_JRA, c= sns.color_palette("deep")[2])

[...]
```

[10]:



Most of it is coherent, except for one noticeable difference with the second decade of ERA, where we can see in the time series (for example at 800hPa) that relative humidity is lower during this period, for a reason we do not know.

2 Line-by-line analysis for Section 3

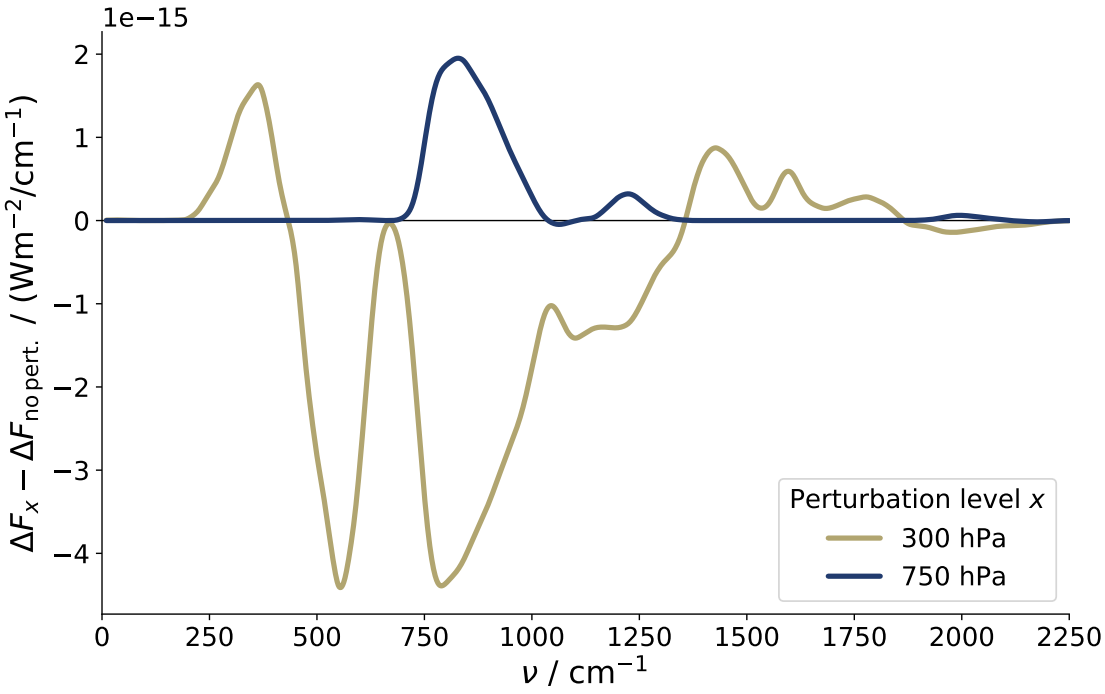


Figure S2. Difference in the "humidity forcing" for RH perturbations at different pressure heights.

Figure S2 shows the change in the "humidity forcing" for relative humidity RH perturbations at different pressure height. We define the humidity forcing as the difference in outgoing-longwave radiation (OLR) between an atmosphere in present-day conditions and an atmosphere in which the absolute humidity has been adjusted to a hypothetically 1-K-warmer temperature profile while preserving the actual temperature. This way, we can quantify the radiative forcing of the moistening of the atmosphere alone. The two lines in Figure S2 show how RH perturbations at different heights affect the humidity forcing: one can see that the change in OLR is increased in spectral regions close to the perturbation. We interpret this as an "anchoring effect" of the perturbation on the effecting emission height z_ϵ. For RH perturbations well above z_ϵ (300 hPa, yellowish line) a stronger increase of the emission height — a stronger forcing — in the atmospheric window overpowers this effect.

3 ECS for different uniform troposheric RH and different surface temperatures

In section 3 (§6) we write that decreasing T_0 reduces the sensitivity to RH.

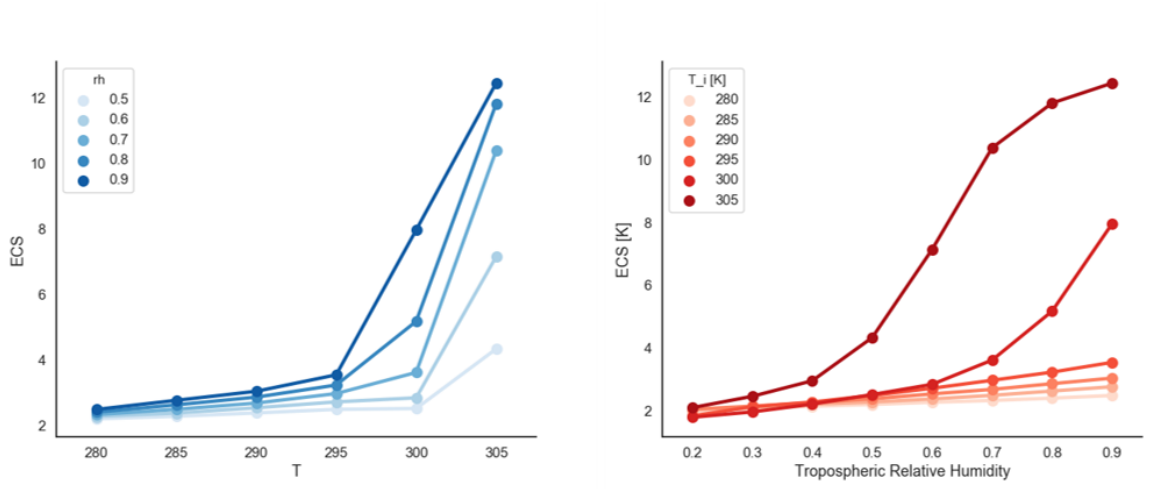


Figure S3. for a set of runs with different initial surface temperature T_0 and different uniform tropospheric RH. All were performed with a moist adiabatic Lapse Rate. Both plots display the same data, but on different axes.

The left panel of Fig. S3 shows this phenomenon. In general, because of the temperature–dependence effect,as highlighted by Meraner, Mauritsen, and Voigt 2013, the atmosphere is less sensitive for lowest temperature, as we can see in left panel of Fig. S3. This effect is even stronger in our case when using $T_0 \geq 300K$, because of the closing of the atmospheric window for such conditions.