s2dverification: Time series visualization function

A. Hunter, J. Giner, N. Manubens

September 27, 2016

1 PlotTimeSeries

This is a preliminary report describing a general purpose time series plotting function in the R package *s2dverification*, and presents several examples that highlight its features. The function, *PlotTimeSeries*, is an agglomerate of the previous time series plotting functions in *s2dverification*, with additional functionality. The function is still evolving, to ensure that it is compatible with the common R data structure which is currently being discussed within the QA4Seas project.

In brief, the *PlotTimeSeries* function takes arrays of any number of (named) dimensions and creates scatter plots or draws curves along them, as well as plotting their means and standard errors. The inputted data can include metadata that is automatically represented on the plot, such as the name of the plotted variable, the start dates of a set of provided seasonal forecasts, the date at each forecast time step or the names of the members. The user can manually specify which dimensions to colour or style along, but the function is designed to anticipate the user's requirements. The advantages of the new function are its increased ease-of-use and versatility, as well as the improved quality of the final plots due to the use of the *ggplot2* package.

2 Examples

This section contains examples of two of the most common formats for a CDM object. In Section 2.1, plots of a CDM object containing temperature anomalies from multiple members of a climate model initialized at different times are presented. Section 2.2 shows examples of a CDM object containing sample statistics (correlation) and their associated confidence intervals.

2.1 Temperature Anomalies

The default settings of the *PlotTimeSeries* function assume that the data to be plotted is either an R object following the common R data structure, and the desired output is a scatter plot with different colours for the start dates and different shapes for the model members, with a smoothing function fitted along the start dates (Loess) and its standard error. However, if these dimensions cannot be automatically found in the data provided, the default settings are disregarded. The function can be adjusted to recognize any custom convention for the dimension names, and also accepts bare R arrays without dimension names or meta-data, allowing for its application to other research fields than climate prediction.
Plots of the mean sea surface temperature anomalies are shown in Fig. 1, using the sample dataset that is loaded with the code in the Appendix.

```
PlotTimeSeries(ano_exp)
```

```
PlotTimeSeries(ano_exp, curves_along = 'member')
PlotTimeSeries(ano_exp, curves_along = NA)
PlotTimeSeries(ano_exp, curves_along = 3)
```

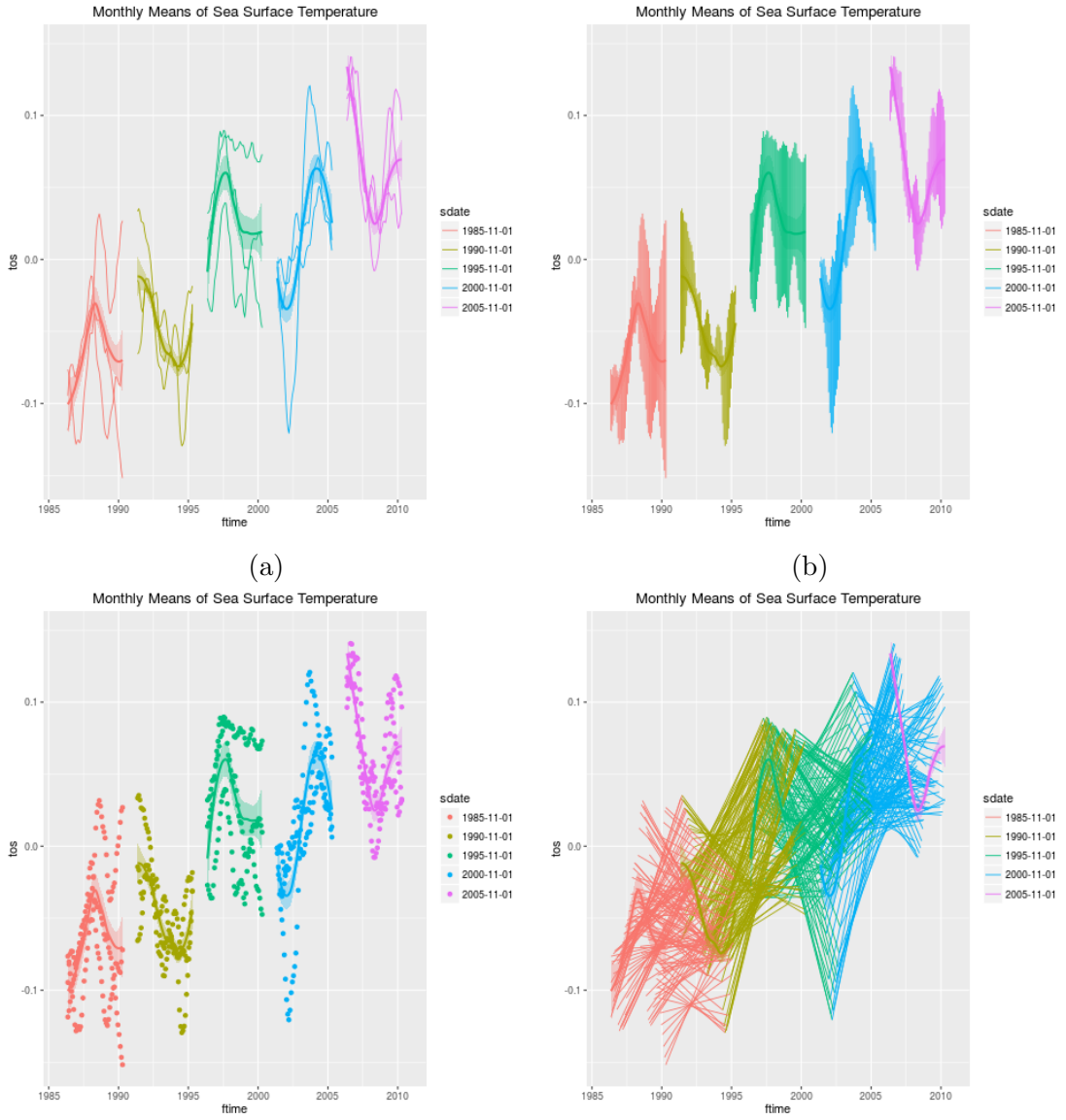
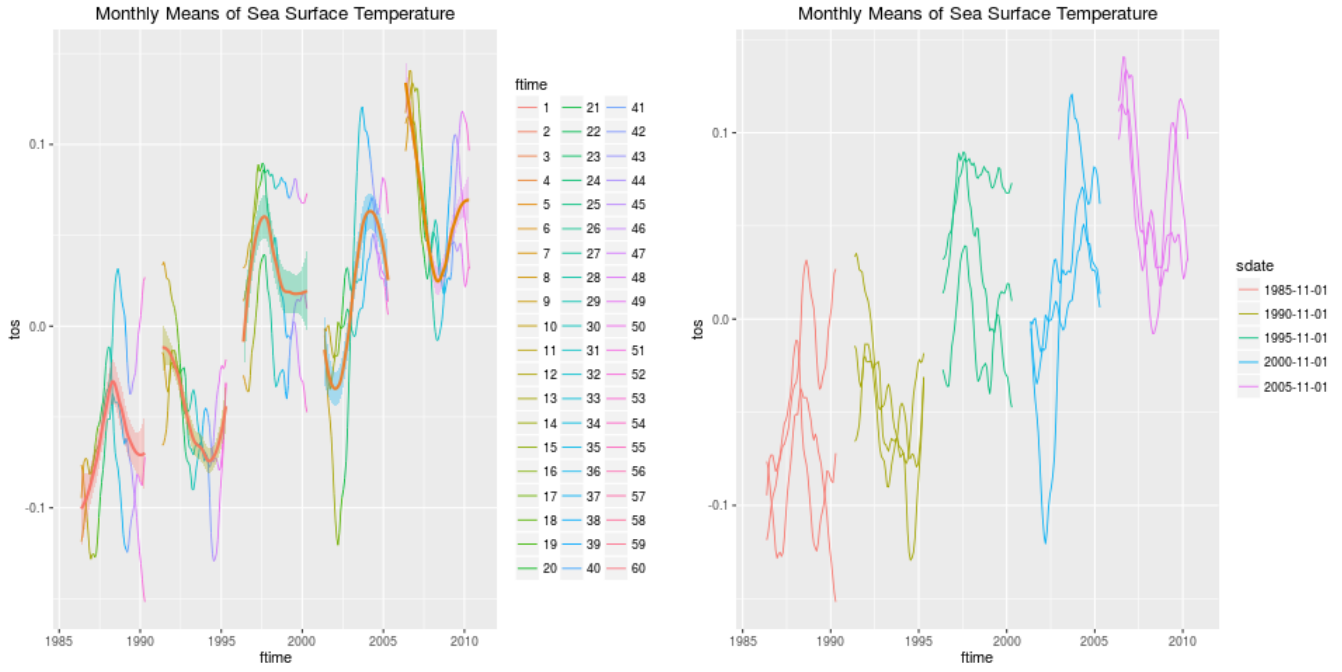


Figure 1: Drift-corrected anomalies of monthly sea surface temperature averages over the whole globe, from the i00k experiment run at BSC-CNS using the EC-Earth (v2.3.0) climate model, initialized every November 1st for the start dates 1985, 1990, 1995, 2000 and 2005. The forecast duration is 5 years

The function allows the user to plot individual model members with or without the mean, or to display only the mean and the maximum and minimum. Although the default setting is for the function to colour along the start dates, the user can change this to any of the named dimensions, as in Fig 2.

```
PlotTimeSeries(ano_exp, colour_along = 'ftime')
PlotTimeSeries(ano_exp, mean_along = NA)
```



(a) (b)

Figure 2: Drift-corrected anomalies of monthly sea surface temperature averages over the whole globe, from the i00k experiment run at BSC-CNS using the EC-Earth (v2.3.0) climate model, initialized every November 1st for the start dates 1985, 1990, 1995, 2000 and 2005. The forecast duration is 5 years

2.2 Correlations

If the input to the function is a CDM object containing a dimension named "test" (as in test statistic), then a plot of these statistics and their associated confidence intervals will automatically be plotted as error bars as in Fig. 3a. Alternatively, the user can specify the confidence intervals to be plotted as dashed lines (and the sample statistic as a solid line) as below.

```
PlotTimeSeries(corr)
PlotTimeSeries(corr, interval_type = "line")
```

(a) (b)

Figure 3: The sample correlation and 95% confidence intervals at each forecast time step of the i00k surface temperature smoothed anomalies with DFSv5.2.

3 Conclusions

The use of the R package *ggplot2* in *PlotTimeSeries* results in high quality time series plots, as well as simplifying the creation of sophisticated graphics. Another benefit of the *ggplot2* package is that the user can easily modify the output of *PlotTimeSeries* to configure the plot title, axis labels and legends. For example, to increase the size of the y axis label in Fig. 3b, and change its content to "Forecast lead time (years)" the below code could be used.

```
cor.plot <- PlotTimeSeries(corr, interval_type = "line")
cor.plot + xlab("Forecast lead time (years)") + theme(axis.title.x
         = element_text(size = rel(1.8), angle = 0))
```

The function is being refined to accommodate a wide range of user requirements. Specifically, the function is being adapted to allow multiple layers to be added to a single plot. This is to allow the plotting of multiple datasets simultaneously, and to interpret metadata, as well as drawing other *ggplot2* plot types. This work is being conducted alongside changes currently being made to the *s2dverification* and *downscaleR* packages to ensure a common R data structure.

4 Appendix

R code for reproducing Figures 1 and 2.

```
# Loading example data
library(s2dverification)
data_path <- system.file('sample_data', package = 's2dverification')
expA <- list(name = 'experiment', path = file.path(data_path,
```

```
'model/$EXP_NAME$/$STORE_FREQ$_mean/$VAR_NAME$_3hourly',
'$VAR_NAME$_$START_DATE$.nc'))
obsX <- list(name = 'observation', path = file.path(data_path,
'$OBS_NAME$/$STORE_FREQ$_mean/$VAR_NAME$',
'$VAR_NAME$_$YEAR$$MONTH$.nc'))

# Ready to use Load().
startDates <- c('19851101', '19901101', '19951101', '20001101', '20051101')
sampleData <- Load('tos', list(expA), list(obsX), startDates,
output = 'areave', latmin = 27, latmax = 48,
lonmin = -12, lonmax = 40)

# Computing bias-corrected anomalies, smoothing them along
# forecast time and computing ensemble mean correlation of
# experimental data with observational data.
clim <- Clim(sampleData$mod, sampleData$obs)
ano_exp <- Ano(sampleData$mod, clim$clim_exp)
ano_obs <- Ano(sampleData$obs, clim$clim_obs)
ano_exp <- Smoothing(ano_exp)
ano_obs <- Smoothing(ano_obs)
corr <- Corr(Mean1Dim(ano_exp, 2), Mean1Dim(ano_obs, 2))

# Transforming to a draft common data structure
ano_exp <- list(Data = ano_exp,
Variable = sampleData$Variable,
Datasets = sampleData$Datasets$exp,
Dates = sampleData$Dates)
class(ano_exp) <- 'multiCDM'

ano_obs <- list(Data = ano_obs,
Variable = sampleData$Variable,
Datasets = sampleData$Datasets$obs,
Dates = sampleData$Dates)
class(ano_obs) <- 'multiCDM'

corr <- list(Data = corr,
Variable = sampleData$Variable,
Datasets = sampleData$Datasets$exp,
Dates = sampleData$Dates)
names(dim(corr$Data))[2] <- 'reference'
names(dim(corr$Data))[3] <- 'test'
corr$Variable$varName <- paste(corr$Variable$varName, "corr")
attr(corr$Variable, 'longname') <- paste(attr(corr$Variable, 'longname'),
                                 "Correlation With", names(ano_obs$Datasets)[1])
corr$Datasets[[1]]$InitializationDates <- NULL
corr$Datasets[[1]]$Members <- NULL
corr$Dates$start = seq(as.POSIXct('00000101', format = '%Y%m%d'),
                       length = 60, by = 'month')
corr$Dates$end = seq(as.POSIXct('00000201', format = '%Y%m%d'),
```

```
                        length = 60, by = 'month')
class(corr) <- 'multiCDM'

PlotTimeSeries(ano_exp)
PlotTimeSeries(ano_exp, curves_along = 'member')
PlotTimeSeries(ano_exp, curves_along = NA)
PlotTimeSeries(ano_exp, curves_along = 3)
PlotTimeSeries(ano_exp, colour_along = 'ftime')
PlotTimeSeries(ano_exp, mean_along = NA)
```

R code for reproducing Figures 3.

```
PlotTimeSeries(corr)
PlotTimeSeries(corr, interval_type = "line")
```