An R Package for Generating Covariance Matrices for Maximum-Entropy Sampling from Precipitation Chemistry Data

Hessa Al-Thani1 · Jon Lee1

Received: 23 December 2019 / Accepted: 20 March 2020 / Published online: 3 August 2020
© Springer Nature Switzerland AG 2020

Abstract

We present an open-source R package (MESgenCov v 0.1.0) for temporally fitting multivariate precipitation chemistry data and extracting a covariance matrix for use in the MESP (maximum-entropy sampling problem). We provide multiple functionalities for modeling and model assessment. The package is tightly coupled with NADP/NTN (National Atmospheric Deposition Program/National Trends Network) data from their set of 379 monitoring sites, 1978–present. The user specifies the sites, chemicals, and time period desired, fits an appropriate user-specified univariate model for each site and chemical selected, and the package produces a covariance matrix for use by MESP algorithms.

Keywords Maximum-entropy sampling · Covariance matrix · Environmental monitoring · Environmetrics · NADP · NTN

1 Introduction

The MESP (maximum-entropy sampling problem) (see [8, 16, 23, 24]) has been applied to many domains where the objective is to determine a “most informative” subset \( Y_S \), of pre-specified size \( s = |S| > 0 \), from a Gaussian random vector \( Y_N \), \( |N| = n > s \). Information is typically measured by (differential) entropy. Generally, we assume that \( Y_N \) has a joint Gaussian distribution with mean vector \( \mu \) and covariance matrix \( C \). Up to constants, the entropy of \( Y_S \) is the log of the determinant of the
principle submatrix $C[S, S]$. So, the MESP seeks to maximize the (log) determinant of $C[S, S]$, for some $S \subseteq N$ with $|S| = s$.

The MESP is NP-hard (see [14]), and there has been considerable work on algorithms aimed at exact solutions for problems of moderate size; see [1–5, 7, 12, 14, 15, 17]. All of this algorithmic work is based on a branch-and-bound framework introduced in [14], and the bulk of the contributions in these references is on different methods for upper bounding the optimal value. This work has been developed and validated in the context of a very small number of data sets, despite the fact that of course multivariate data is widely available. The reason for this shortcoming is that despite all of the raw multivariate data that is available, it is not a simple matter to turn this data into meaningful covariance matrices for Gaussian random variables.

Our goal with the R package (MESgenCov v 0.1.0) that we have developed is to provide such a link — between readily available raw environmental-monitoring data and covariance matrices suitable for the MESP — in the context of environmental monitoring. Our work fits squarely into recent efforts to better exploit massive amounts of available data for mathematical-programming approaches to decision problems. Even if we have reliable raw data, we can only make good decisions if we have a means to prepare that data so that we can populate our mathematical-programming models in such a way as to meet required assumptions.

We note that another R package of interest is [19]: “EnviroStat provides functions for spatio-temporal modeling of environmental processes and designing monitoring networks for them based on an approach described in [18]”.

In Section 2, we discuss application of the MESP to environmental monitoring and the NADP/NTN (National Atmospheric Deposition Program / National Trends Network) data environment. In Section 3, we describe our methodology. In Section 4, we describe our R package (MESgenCov v 0.1.0). In Section 5, we make some concluding remarks.

2 Environmental Monitoring and NADP/NTN Data

A key area of application for the MESP has been in environmental monitoring (see [6, 9, 25], for example). The idea is that precipitation is collected at many sites, and its chemistry is analyzed. This is costly, and it is a natural question as to whether a subset of the sites might yield data without much loss of information (as measured by entropy). But it is a challenge to process the raw data in such a way that multivariate normality is achieved because only then are the model of the MESP and its related algorithms applicable.

The NADP maintains the NTN (see [21]); this network measures the chemistry (i.e., ammonium, calcium, chloride, hydrogen, magnesium, nitrate, pH, potassium, sodium, and sulfate) of precipitation at 379 monitoring sites across the USA, with some data available as far back as 1978; at present, 255 sites are active.

Our R package is tightly coupled with this precipitation and chemistry data. We are interested in instances of the MESP where $n$ user-specified site/chemical pairs
comprise $N$. Precipitation data (measured in L) are available on a daily basis, and chemical concentrations (measured in mg/L) are available on a weekly basis. These datasets are available in the packages `EnvMonDataConc` and `EnvMonDataPre`. They can be installed and loaded using the `devtools` library. If not already installed, the `devtools` library can be installed and loaded as follows:

```
#Install and load devtools
> install.packages("devtools")
> library(devtools)
```

Then, we can install the data packages (from GitHub) and load them:

```
#Install data packages
> install_github("hessakh/EnvMonDataConc")
> install_github("hessakh/EnvMonDataPre")

#Load data packages
> library(EnvMonDataConc)
> data("weeklyConc")
> library(EnvMonDataPre)
> data("preDaily")
```

A full description of the daily and weekly precipitation data appears in Appendix A, derived from http://nadp.slh.wisc.edu/data/ntn/meta/ntn-daily-Meta.pdf and http://nadp.slh.wisc.edu/data/ntn/meta/ntn-weekly-Meta.pdf, courtesy of the NADP.

Small snapshots of the data can easily be viewed. For example, we can output the first 6 rows and first 5 columns of the weekly raw data.

```
#Display part of the weeklyConc data frame
weeklyConc[1:6,1:5]

   siteID dateon   dateoff yrmonth    ph
 1     AB32 18:40:00 15:10:00 201609 -9.00
 2     AB32 15:15:00 16:00:00 201609 -9.00
 3     AB32 16:00:00 16:55:00 201610  6.56
 4     AB32 16:55:00 17:00:00 201610 -9.00
 5     AB32 17:00:00 20:00:00 201610 -9.00
 6     AB32 20:00:00 18:00:00 201610  4.73
```

Note that missing concentration values are coded as -9.00.

---

1National Atmospheric Deposition Program (NRSP-3). 2019. NADP Program Office, Wisconsin State Laboratory of Hygiene, 465 Henry Mall, Madison, WI 53706, USA
3 Our Methodology

3.1 NADP/NTN Data Processing

We process the raw NADP/NTN data in a similar way to earlier uses in the context of the MESP in the field of environmental statistics (see [9]).

We calculate the level of a chemical’s concentration by summing weekly quantities (mg) of the chemical, over a month, and dividing the monthly total by total precipitation (L), over dates in that month, to get monthly values of chemical concentrations (mg/L). We use monthly concentrations instead of the available weekly concentrations because there is a large proportion of missing data for individual weeks compared with full months. Furthermore, the univariate models were better at predicting average monthly concentrations than they were at predicting weekly concentrations.

For a given monitoring site, chemical, and month \( t = 0, 1, \ldots, T - 1 \), let

\[
\begin{align*}
W(t) & := \text{set of weeks in month } t, \\
D(w) & := \text{set of days in week } w, \\
c_w & := \text{recorded chemical concentration (mg/L) for week } w \\
& \quad (c_w = \ast \text{ denotes an unrecorded value}), \\
p_d & := \text{recorded precipitation quantity (L) for day } d, \\
p_w & := \text{precipitation quantity (L) for week } w; \\
& \quad p_w = \sum_{d \in D(w)} p_d.
\end{align*}
\]

Then, the chemical concentration (mg/L) for month \( t \) is calculated as:

\[
y(t) := \frac{\sum_{w \in W(t):c_w \neq \ast} p_w c_w}{\sum_{w \in W(t):c_w \neq \ast} \sum_{d \in w} p_d}.
\]

It should be noted that when there is no weekly value available for the chemical quantity, we do not use the precipitation values for any of the days in such a week (so as to not artificially dilute the chemical concentration level for the month).

Next, we fit a temporal model to \( \log(y(t)) \), which is a rather standard method for handling heavy-tailed distributions. We note that our data has no zero concentrations, so there is no issue of \( \log(0) \).

A quick look at some graphics indicates that there are clear long-term trends (see Fig. 1\textsuperscript{2}), from which we can see that sulfate concentrations are generally trending downward over time. Again, looking at some data, we can easily see periodic trends (see Fig. 2), where we can easily see a yearly periodicity.

The general model that we provide is:

\[
\log(y(t)) = \sum_{i=0}^{r} \beta_i t_i^i + \sum_{j=1}^{k} a_j \cos \left( \frac{2\pi j t}{12} \right) + b_j \sin \left( \frac{2\pi j t}{12} \right),
\]

with the parameters \( \beta_i, a_j, \) and \( b_j \) fit by ordinary linear regression. The user can specify the degree \( r \) for the polynomial part of the model which we think of as a truncated

\textsuperscript{2}Reprinted with the kind permission of the National Atmospheric Deposition Program (NRSP-3). 2019. NADP Program Office, Wisconsin State Laboratory of Hygiene, 465 Henry Mall, Madison, WI 53706, USA.
Taylor series, aimed at capturing aperiodic trends. Periodic trends are captured via a truncated Fourier series, truncated at level $k$.

We note that [9] used the following model to de-seasonalize and de-trend the log-transformed monthly sulfate concentration values:

$$\log(y(t)) = \beta_1 + \beta_2 t + a_1 \cos \left( \frac{2\pi t}{12} \right) + b_1 \sin \left( \frac{2\pi t}{12} \right).$$

Fig. 1  Sulfate concentration over time

Fig. 2  Log sulfate concentration over a 4-year period at a site
This is just an affine model $\beta_1 + \beta_2 t$ plus a sinusoidal model with monthly periodicity and intercept $a_1$. The simple model Eq. 2 is Eq. 1 with $r = 1$ and $k = 1$. We found that Eq. 2 did well at normalizing the errors for certain sites, but for some sites, such as “AL10” and “IN41,” Eq. 2 did not do so well. So rather than fix Eq. 2 as the model for our R package, we provide the flexibility of Eq. 1.

In Fig. 3, we provide an example where Eq. 1 with $r = 1$ and $k = 3$ is a much better fit for the log concentration of sulfate at site “TN00” than Eq. 2. The plot on the left shows the fit of model Eq. 2, and the plot on the right shows the fit of our model Eq. 1.

To produce the covariance matrix, we need error values for each time point. Missing values in the NADP/NTN dataset mean that each set of error values produced by the model may vary in size. So we have filled in missing values for each site and time point by sampling from a normal distribution with the mean being the predicted value by the univariate model and the standard deviation being the standard error of the univariate model.

### 4 MESgenCov

Our R package MESgenCov can be obtained from [https://github.com/hessakh/MESgenCov](https://github.com/hessakh/MESgenCov) and installed using the devtools library.

```r
# Install and load MESgenCov
> install_github("hessakh/MESgenCov")
> library(MESgenCov)
```

MESgenCov contains functions in the S3 class to create a covariance matrix from the desired subset of NADP/NTN data. The function `getCov()` returns a covariance matrix, a list of univariate model summaries, and a table of normality tests produced
by the MVN R package (see [13]). The multivariate analysis is used to assess the validity of the covariance matrix to be used as input for the MESP. The user can make adjustments to the input of `getCov()` so as to obtain a covariance matrix for a multivariate Gaussian vector, which is then valid for use in the MESP.

To avoid sites with a small sample size for the specified time frame, the function `getSites()` outputs a vector of the sites with the largest sample of data for a given time frame and measured chemical (see Section 4.2). To find sites that are spatially “spread out” but have at least some specified sample size, the function `maxDistSites()` can be used to obtain a list of geographically sparse sites (see Section 4.2). Finally, in the case where the residuals from a univariate model do not appear to be normally distributed, the function `lambertWtransform()` allows the user to transform the residuals (from a univariate model) using the R package LambertW: Probabilistic Models to Analyze and Gaussianize Heavy-Tailed, Skewed Data (see [11]). This can be very effective in situations where the distributions seem to have heavy tails and some skewness (see [10] and Section 4.3).

### 4.1 `getCov`

`getCov()` takes a 14-column data frame as input where each column corresponds to one of the user specifications shown in Fig. 4. The 14 specifications in the input allow the user to specify the subset of data to analyze and gives the user options in displaying different parts of the analysis.

#### 4.1.1 Input

A default set of inputs can be found in the stored data frame “defaultInput.” Each column of “defaultInput” is an argument in the function `getCov()`. After storing “defaultInput” in a variable in the user’s workspace, the input can be changed. For example, below we store the “defaultInput” data frame in a variable “df,” and then change the end date:

```r
# Load defaultInput data frame and store in df
> data("defaultInput")
> df <- defaultInput
> df

  startdateStr enddateStr use36 siteAdd
1  01/01/83 00:00 12/31/86 00:00 TRUE NULL

  outlierDatesBySite siteOutliers comp plotMulti sitePlot
1 NULL          NULL S04 FALSE NULL

  plotAll writeMat r k
FALSE FALSE 1 1

# Change the end date to extend the sample of data taken from weeklyConc
df$enddateStr <- "12/31/88 00:00"
```
### Arguments

| Argument           | Definition                                                                 |
|--------------------|---------------------------------------------------------------------------|
| startdateStr       | Date and time of when to start analyzing the data, in the format = m/d/y H:M |
| enddateStr         | Date and time of when to stop analyzing the data, in the format = m/d/y H:M |
| comp               | String of pollutant or acidity level to be analyzed, the pollutants name should be used as it appears in weeklyConc |
| use36              | TRUE if default 36 sites should be added, FALSE otherwise                  |
| siteAdd            | List of strings of siteIDs that should be analyzed                          |
| outlierDatesbySite | List of sites where outliers should be analyzed                             |
| siteOutliers       | List of sites where outliers should be removed                              |
| removeOutliers     | Specify siteID string for outlier analysis                                  |
| plotMulti          | TRUE if multivariate analysis plots should be displayed, FALSE otherwise   |
| sitePlot           | Specify list of siteIDs to be plotted                                       |
| plotAll            | TRUE if plots for all sites should be displayed, FALSE otherwise           |
| writeMat           | TRUE if .mat file of the resulting covariance matrix should be written in the working directory |
| r                  | Integer <= 5, see univariate model                                           |
| k                  | Integer <= 5, see univariate model                                           |

Fig. 4 Input parameters for `getCov()`

### 4.1.2 Output

The function `getCov()` produces a list with the following elements:

| Output        | Definition                                                                                           |
|---------------|-----------------------------------------------------------------------------------------------------|
| cov           | Covariance matrix produced by univariate model residuals                                              |
| listMod       | List of univariate model summaries produced by `lm()`                                                 |
| sites         | List of sites that were analyzed                                                                    |
| mvn           | Output of the MVN package                                                                            |
| univariateTest| Univariate test output, also by the MVN package                                                      |
| residualData  | Data frame of residuals produced by the univariate model                                             |
| residualDataNA| Data frame of residuals, where missing values are left as NA                                         |
| rosnerTest    | Output of the Rosner test for outlier analysis produced by the EnvStats package; see [20]           |
| pred          | List of predicted values produced by the univariate model for each site                              |

Next, we demonstrate how to access these elements and certain plots after running the function `getCov()`.
1. Multivariate and univariate normality

```r
# Change part of the input data frame df
> df$plotMulti <- TRUE
# Change univariate model parameter from 1 to 3
> df$k <- 3
# Store output in variable g so that the list of outputs given by getGcov() can be called
> g <- getGcov(df)
```

![Chi-Square Q-Q Plot](image)

```r
> g$univariateTest

| Test          | Variable | Statistic | p value | Normality |
|---------------|----------|-----------|---------|-----------|
| 1 Shapiro-Wilk| AL10S04  | 0.9347    | 0.001   | NO        |
| 2 Shapiro-Wilk| IL11S04  | 0.9894    | 0.8121  | YES       |
| 3 Shapiro-Wilk| IL18S04  | 0.9909    | 0.8854  | YES       |
| 4 Shapiro-Wilk| IL19S04  | 0.9184    | 2e-04   | NO        |
| 5 Shapiro-Wilk| IL35S04  | 0.8709    | <0.001  | NO        |
```
2. Outlier test for specific tests

```r
df$siteOutliers <- list(c("IN41"))
df$sitePlot <- list(c("IN41"))
g <- getCov(df)
i <- match("IN41",g$sites)
g$rosnerTest[[i]]$all.stats

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| i | Mean.i | SD.i | Value | Obs.Num | R.i+1 | lambda.i+1 | Outlier |
| 1 | 0.0069  | 0.2814 | 0.9359 | 25      | 3.3013 | 3.2680     | TRUE    |
| 2 | 0.0062  | 0.2604 | 0.7194 | 30      | 2.7862 | 3.2628     | FALSE   |
| 3 | 0.0165  | 0.2471 | 0.7215 | 66      | 2.8533 | 3.2576     | FALSE   |
```

By changing the input, we can remove the outliers detected by the Rosner test. Note that the plots are generated after running `getCov()`. Furthermore, `getCov()` does not need to be stored in a variable to generate the plots.

```r
# Remove month 25 from site IN41's pollutant concentration data
> df$outlierDatesBySite <- c("IN41",25)
> getCov(df)
```
3. Outlier test for all sites

```r
# take sites used in analysis in g and run outlier test
df$siteOutliers <- list(g$sites)

# Remove data points identified as outliers from these sites
df$removeOutliers <- list(g$sites)
g <- getCov(df)
```

4. Plot all sites

```r
> df$plotAll <- TRUE
> getCov(df)
```
5. Output all MVN package analysis

The following output is from a call to the MVN package that produces multivariate analysis based on the Mardia method and univariate analysis based on the Shapiro-Wilson method.

```r
# Remove default list of sites so that their data is not analyzed
# Add new site list
> df$use36 <- FALSE
> df$siteAdd <- list(c("NY52", "TN11", "IL63"))
# Remove any set of sites and pollutant combinations that had been previously added
> df$siteOutliers <- NULL
> df$outlierDatesBySite <- NULL
> df$removeOutliers <- NULL
> g <- getCov(df)
# Display full output of the MVN package
> g$mvn

$multivariateNormality
Test Statistic p value Result
1 Mardia Skewness 14.019 0.1721 YES
2 Mardia Kurtosis -0.1465 0.8835 YES
3 MVN <NA> <NA> YES

$univariateNormality
Test Variable Statistic p value Normality
1 Shapiro-Wilk NY52S04 0.9821 0.3981 YES
2 Shapiro-Wilk TN11S04 0.9830 0.4385 YES
3 Shapiro-Wilk IL63S04 0.9873 0.6858 YES

$Descriptives
n Mean Std.Dev Median Min Max
NY52S04 72 1.1709e-17 0.2812 0.0333 -0.8815 0.6170
TN11S04 72 -1.4991e-02 0.4666 -0.0020 -0.9826 1.4243
IL63S04 72 4.8127e-18 0.3016 -0.0373 -0.7353 0.7899
25th 75th Skew Kurtosis
NY52S04 0.2211 0.1840 -0.3702 0.0894
TN11S04 -0.326 0.2130 0.4514 0.2981
IL63S04 -0.18369 0.1724 0.3287 0.0609
```
The specific call of the MVN package is

```r
> mvn(dfRes[,1], subset = NULL, mvnTest = "mardia",
  covariance = TRUE, tol = 1e-25, alpha = 0.5, scale = FALSE,
  desc = TRUE, transform = "none", univariateTest = "SW",
  univariatePlot = "none", multivariatePlot = "none",
  multivariateOutlierMethod = "none", bc = FALSE, bcType =
  "rounded", showOutliers = FALSE, showNewData = FALSE).
```

See [13] for details on the MVN package.

6. Covariance matrix

```r
> g <- getCov(df)
# Print covariance matrix
> round(g$ cov, digits = 4)
```

|        | NY52S04 | TN11S04 | IL63S04 |
|--------|---------|---------|---------|
| NY52S04| 0.0791  | 0.0047  | 0.0009  |
| TN11S04| 0.0047  | 0.2177  | 0.0185  |
| IL63S04| 0.0009  | 0.0185  | 0.0909  |

7. Save covariance matrix as a .mat file (to populate an instance of the MESP, for example).

This is done by simply setting the input data frame attribute `writeMat` to TRUE. The .mat file will be saved to the user’s current working directory as `covSites.mat`. For processing further with Matlab, use the (Matlab) “load” command.

```r
# Write cov into .mat file in current directory
> df$writeMat <- TRUE
> g <- getCov(df)
```

In the case that the user has already generated an output by the function `getCov()`, it is possible to also create the .mat file in the following manner.

```r
> library(rmatio)
> write.mat(g$ cov, filename = "covariance1.mat")
```
8. Univariate model summaries

```r
> result <- getGon(df)
> sites <- result$sites
> i = match(c("NY52"), sites)
> result$listMod[i]

[[1]]
Call:
  lm(formula = y1 ~ I(cos(t*(2*pi/s))) + I(sin(t*(2*pi/s))) + I(cos(t*(2*pi/s)*2)) + I(sin(t*(2*pi/s)*2)) + I(cos(t*(2*pi/s)*3)) + I(sin(t*(2*pi/s)*3)) + I(t), data = df)

Residuals:
   Min     1Q Median     3Q    Max
-0.5236 -0.1677 -0.0189  0.1818  0.9087

Coefficients:     Estimate Std. Error t value Pr(>|t|)
(Intercept)      1.1110     0.0712   15.40  < 2e-16 ***
I(cos(t*(2*pi/s))) -0.3541     0.0494   -7.14   1.0e-10 ***
I(sin(t*(2*pi/s))) -0.1109     0.0498   -2.22     0.041 *
I(cos(t*(2*pi/s)*2))  0.0500     0.0494    1.01     0.315
I(sin(t*(2*pi/s)*2))  0.0797     0.0494    1.61     0.110
I(cos(t*(2*pi/s)*3))  0.0391     0.0494   -0.80     0.416
I(sin(t*(2*pi/s)*3))  0.0536     0.0494   -1.08     0.284
I(t)                -0.0010     0.0017   -0.61     0.540
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2961 on 64 degrees of freedom
Multiple R-squared:  0.6265,    Adjusted R-squared:  0.5857
F-statistic: 15.34 on 7 and 64 DF,  p-value: 1.327e-11
```
9. Output data frame of residuals

```r
> g <- getGcov(df)
#Display dataframe containing the residuals
from the fitted univariate model
> g$residualData[1:5,]

   NY52S04  TN11S04  IL63S04
1 0.1959813 -0.43477735 0.2824803
2 0.6170340 -0.38791938 -0.3495080
3 -0.4510128  1.05519620 -0.1158200
4 -0.1580145 -0.01789642 -0.1297776
5 -0.3218159 -0.47833685 -0.3415119
```

4.2 Functions for Getting a Vector of Sites

`getGcov()` takes site lists as input. The function `getSites()` produces a list of sites with available data for a specified time frame. The code below produces a list of 36 sites with the most weekly data between the years 1983 and 1986.

```r
> result <- getSites("01/01/83 00:00","12/31/86 00:00",36,104,
                   "SO4",""
> result$finalList

[1] "OH71" "NY08" "WV18" "MI53" "NH02" "OH49" "PA42" "ME09"
[9] "IN34" "MA13" "NY52" "NY10" "WA14" "NY20" "OH17" "ME00"
[17] "TN00" "IL63" "MI99" "WI28" "IN41" "PA29" "WI36" "ME02"
[25] "MI09" "MO05" "NC03" "NJ99" "PA15" "CO19" "MN18" "WI37"
[33] "AR27" "KS31" "ME98" "MO03"
```

The 4th input specifies the minimum sample of weekly data required to be included in the produced list, and the last input tells the function to only look at sites in the Northern region of the USA. The options for region are “W,” “S,” and “N”; see Appendix B for the geographic split.

```r
> NSites <- getSites("01/01/83 00:00","12/31/86 00:00",36,104,
                   "SO4","N"
> NSites$finalList

[1] "OH71" "NY08" "MI53" "NH02" "OH49" "PA42" "ME09" "IN34"
[9] "MA13" "NY52" "NY10" "NY20" "OH17" "ME00" "MI99" "WI28"
[17] "IN41" "PA29" "WI36" "ME02" "MI09" "NJ99" "PA15" "MN18"
[25] "WI37" "ME98" "IL11" "IL18" "MN16" "MI26" "NE15" "VT01"
[33] "NY99" "MA01" "MA08" "MN27"
```

The function `maxDistSites()` prioritizes sites that are farther away from each other. This function takes the same arguments as input as `getSites()`, except for the last argument where instead of specifying a region, the user can specify which site should be included first. If the last argument is 1, then the site with the greatest
amount of data for the specified time period will be chosen; if the last argument is 2, then the site with the second greatest amount of data will be chosen; etc.

```r
> maxdist <- maxDistSites("01/01/83 00:00","12/31/86 00:00",36,
                          104,"SD4",1)
> maxdist$finalList

[1] "OH71" "WA14" "TX04" "FL11" "ME00" "WY06" "MN27" "LA12"
[9] "CA45" "OK00" "NY99" "GA41" "MI99" "AZ03" "MT05" "NC35"
[17] "MO05" "CO00" "WY99" "IN34" "KY03" "MI09" "FL03" "MA01"
[25] "OR10" "PA42" "AR27" "MN16" "TX21" "VT99" "NE15" "VA13"
[33] "CO15" "CO22" "NY52" "AR02"
```

### 4.3 Lambert W Transformation on Univariate Data

For a number of sites, the residuals produced by our univariate model have skewed distributions with heavy tails. In particular, this is the case for many sites when the sample of data is taken over a period greater than 4 years. To deal with this issue, we have incorporated functions from the LambertW package (see [11]) in the function `lambertWtransform()` that will allow a user to transform the residuals produced by the deterministic univariate model. The LambertW package estimates the parameters that fit a Lambert W distribution on the given univariate data. Then, the underlying Gaussian distribution implied by the Lambert W distribution is extracted and is used for the multivariate analysis in the function `lambertWtransform()`. The `lambertWtransform()` function takes the following as input: a data frame of residuals, and two logical inputs specifying whether to plot the multivariate qq plot and whether to produce the .mat file containing the covariance matrix with the Lambert W transformed residuals. Details on the algorithms that perform the transformation can be found in [10]. Here, we provide an example where we transform the residuals of 50 sites stored in an internal dataset, named `dfRes50`.

```r
> data("dfRes50")
> loutput <- lambertWtransform(dfRes=dfRes50, plotMulti=FALSE,
                               writeMat=FALSE)
> loutput$mvn$multivariateNormality

| Test    | Statistic | p value | Result |
|---------|-----------|---------|--------|
| 1 Mardia Skewness | 22800 | 0.0004 | NO     |
| 2 Mardia Kurtosis  | 0.418  | 0.6763 | YES    |
| 3 MVN              | <NA>   | <NA>   | NO     |
```

This function produces a list of four outputs:

1. `loutput$mvn` contains the results of applying the multivariate analysis by the MVN package
2. `loutput$cov` contains the covariance matrix produced by the transformed residuals
3. `loutput$newResiduals` contains the data frame of Lambert W transformed residuals
4. `loutput$univariateTest` contains the univariate tests produced by the MVN function for the transformed residuals

```r
> data("dfRes50")
> dfRes50 <- dfRes50
> loutput <- lambertWtransform(dfRes=dfRes50, plotMulti=FALSE, writeMat=FALSE)
> loutput$mvn
> loutput$cov
> loutput$newResiduals
> loutput$univariateTest
```

Next, we present an example where we use `maxDistSites()` to get a list of 50 sites that is geographically sparse and has at least 200 weeks of data between 1986 and 1994. From this list of sites, a covariance, and its corresponding multivariate normality test, is generated and compared with the Lambert W transformed output.

```r
#Get list of sites
> maxd <- maxDistSites("01/01/86 00:00","12/31/94 00:00",50, 200,"SO4",1)

#Create input data frame
> df <- defaultInput

#Use list of sites and specification in maxd
> df$siteAdd <- list(maxd$finalList)
> df$startdateStr <- maxd(startDate
> df$use36 <- FALSE
> df$comp <- maxd$comp
> df$enddateStr <- maxd$endDate
> df$writeMat <- TRUE
> output <- getCov(df)
> output$mvn$multivariateNormality

| Test        | Statistic | p value  | Result |
|-------------|-----------|----------|--------|
| 1 Mardia Skewness | 22962 | 2.6e-05 | NO     |
| 2 Mardia Kurtosis   | 0.2408 | 0.8097 | YES    |
| 3 MVN             | <NA>    | <NA>    | NO     |
```

```r
loutput <- lambertWtransform(g$residualDataNA, TRUE,FALSE)
loutput$mvn$multivariateNormality

| Test        | Statistic | p value  | Result |
|-------------|-----------|----------|--------|
| 1 Mardia Skewness | 22263 | 0.2174 | YES    |
| 2 Mardia Kurtosis   | -1.6039 | 0.1087 | YES    |
| 3 MVN             | <NA>    | <NA>    | YES    |
```
4.4 Internal Datasets and Their Properties

We provide five internal covariance matrices that we have produced from geographically sparse lists of sites. The exact specifications used to produce the sites can be found in Appendix C. We offer these covariance matrices for the convenience of the user. We note that although the sites in the list are quite spread out geographically, they are not independent. We test the independence of the covariance matrices using a likelihood ratio test (see [22, p. 275]).

The test statistic is:

\[ u := - \left( \nu - \frac{2m + 5}{6} \right) \log(\det(R)), \]

where \( m \) := number of sites, \( \nu := m(m + 1)/2 \), and \( R \) is the sample correlation matrix. The null hypothesis \( H_0 \) is that the variates are independent, and we reject \( H_0 \) if \( u > \chi^2_{m(m-1)/2, \alpha} \) where for our analysis \( \alpha = 0.05 \). The five covariance matrices are named “maxd1Cov,” “maxd2Cov,” “maxd3Cov,” “maxd4Cov,” and “maxd5Cov,” and the corresponding test statistics \( u \) are 21605, 25740, 26928, 25870, and 25137, all comfortably giving evidence to reject \( H_0 \) at the \( \alpha = 0.05 \) level (we reject when \( u > 1308 \)).

We have made a function available that performs this independence test. Here, we indicate how it can be used on a data frame of residuals produced by \texttt{getCov()}.

```r
> maxd1 <- maxDistSites("01/01/86 00:00","12/31/94 00:00",50,
| 250,"S04",1)
> df$comp <- maxd1$comp
> df$endateStr <- maxd1$endDate
> df$startdateStr <- maxd1$startDate
> df$sitesAdd <- list(maxd1$finalList)
> result <- getCov(df)
> indp <- independenceTest(result$residualData)
> indp$test

chisq dist likelihood ratio tchisq independent
 1       21605  1307.54     FALSE
```

5 Concluding Remarks

We are currently working on enhancements to \texttt{MESgenCov}. Ultimately, we would like to make it easy to use datasets from other application domains, and to make it easier for a user to use other univariate models than the one we provide. Finally, we hope to eventually have a seamless integration with algorithms for the MESP.

Acknowledgments The authors are very grateful to Dr. Martin Shafer and Robert Larson for helping us gain access to the NADP/NTN data in a convenient form.
**Funding Information** J. Lee was funded by the Air Force Office of Scientific Research (Complex Networks program), FA9550-19-1-0175. H. Al-Thani was funded by the Qatar National Research Fund (Graduate Sponsorship Research Award), GSRA4-2-0526-17114.

**Compliance with Ethical Standards**

**Conflict of Interest** The authors declare that they have no conflict of interest.

**Appendix**

**A NADP/NTN Data Descriptions**

### NADP/NTN Daily Data

| Column number | Field    | Data type | Description                                                                 |
|---------------|----------|-----------|-----------------------------------------------------------------------------|
| 1             | SiteID   | Char(4)   | Site Identifier                                                             |
| 2             | StartTime| Char(16)  | Period start, reported in Greenwich Mean Time (GMT) YYYY-MM-DD hh:mm format |
| 3             | EndTime  | Char(16)  | Period end, reported in Greenwich Mean Time (GMT) YYYY-MM-DD hh:mm format   |
| 4             | Amount   | Integer   | Precipitation depth, inches Missing = -9, Trace precipitation amount = -7    |

### NADP/NTN Weekly Data

| Column number | Field      | Data type | Description                                                                 |
|---------------|------------|-----------|-----------------------------------------------------------------------------|
| 1             | SiteID     | Char(4)   | Site Identifier                                                             |
| 2             | DateOn     | Char(16)  | Date on which the sample bucket was installed on the collector, reported in Greenwich Mean Time (GMT) YYYY-MM-DD hh:mm format |
| 3             | DateOff    | Char(16)  | Date on which the sample bucket was removed from the collector, reported in Greenwich Mean Time (GMT) YYYY-MM-DD hh:mm format |
| 4             | yrMonth    | Integer   | Year and Month of sample midpoint, in YYYYMM format                           |
| 5             | pH         | Decimal   | Negative log of the hydrogen ion concentration as measured at the CAL, in pH units |
| 6             | Ca         | Decimal   | Ca concentration, mg/L                                                       |
| 7             | Mg         | Decimal   | Mg concentration, mg/L                                                       |
| 8             | K          | Decimal   | K concentration, mg/L                                                        |
| 9             | Na         | Decimal   | Na concentration, mg/L                                                       |
| 10            | NH4        | Decimal   | NH4 concentration, mg/L                                                      |
| 11            | NO3        | Decimal   | NO3 concentration, mg/L                                                      |
| 12            | Cl         | Decimal   | Cl concentration, mg/L                                                       |
| 13            | SO4        | Decimal   | SO4 concentration, mg/L                                                      |
| 14            | Br         | Decimal   | Br concentration, mg/L                                                       |
B Geographic Split

C Internal Covariance Matrices Site Lists

```r
# Sites with maximum distance data sets, get 50 sites
> maxd1 <- maxDistSites("01/01/86 00:00","12/31/94 00:00",50,200,"SO4",1)
> maxd2 <- maxDistSites("01/01/07 00:00","12/31/14 00:00",50,230,"SO4",1)
> maxd3 <- maxDistSites("01/01/07 00:00","12/31/14 00:00",50,230,"NO3",1)
> maxd4 <- maxDistSites("01/01/07 00:00","12/31/14 00:00",50,230,"Na",1)
> maxd5 <- maxDistSites("01/01/07 00:00","12/31/14 00:00",50,230,"ph",1)
```

References

1. Anstreicher KM, Fampa M, Lee J, Williams J (1999) Using continuous nonlinear relaxations to solve constrained maximum-entropy sampling problems. Math Program 85(2, Ser. A):221–240
2. Anstreicher KM, Lee J (2004) A masked spectral bound for maximum-entropy sampling. In mODa 7—advances in model-oriented design and analysis, Contrib. Statist., pages 1–12. Physica, Heidelberg
3. Anstreicher KM (2018) Efficient solution of maximum-entropy sampling problems Preprint available at: https://www.biz.uiowa.edu/faculty/anstreicher/papers/linx.pdf
4. Anstreicher KM (2018) Maximum-entropy sampling and the boolean quadric polytope. J Glob Optim 72(4):603–618
5. Burer S, Lee J (2007) Solving maximum-entropy sampling problems using factored masks. Math Program 109(2-3, Ser. B):263–281
6. Brown PJ, Le ND, Zidek JV (1994) Multivariate spatial interpolation and exposure to air pollutants. Canad J Statist 22(4):489–509
7. Chen Z, Fampa M, Lambert A, Lee J (2020) Mixing convex-optimization bounds for maximum-entropy sampling. arXiv:2001.11896
8. Fedorov V, Lee J (2000) Design of experiments in statistics. In: Handbook of semidefinite programming, vol 27 of Internat. Ser. Oper. Res. Management Sci., pages 511–532. Kluwer Acad. Publ., Boston, MA
9. Guttrop P, Le ND, Sampson PD, Zidek JV (1993) Using entropy in the redesign of an environmental monitoring network. In: Multivariate environmental statistics, vol 6 of North-Holland Ser. Statist. Probab., pp 175–202. North-Holland, Amsterdam
10. Goerg GM (2011) Lambert W random variables - a new family of generalized skewed distributions with applications to risk estimation. Ann Appl Stat 5(3):2197–2230
11. Goerg GM (2016) LambertW: Probabilistic models to analyze and gaussianize heavy-tailed, skewed data. Version 0.6.4. https://cran.r-project.org/web/packages/LambertW/
12. Hoffman AJ, Lee J, Williams J (2001) New upper bounds for maximum-entropy sampling. In: mODa 6—advances in model-oriented design and analysis (Puchberg/Schneeberg, 2001), Contrib. Statist., pages 143–153. Physica, Heidelberg
13. Korkmaz S, Goksuluk D, Zararsiz G (2014) MVN: An R package for assessing multivariate normality. R J 6(2):151–162
14. Ko C-W, Lee J, Queyranne M (1995) An exact algorithm for maximum entropy sampling. Oper Res 43(4):684–691
15. Lee J (1998) Constrained maximum-entropy sampling. Oper Res 46(5):655–664
16. Lee J (2012) Encyclopedia of environmetrics. In: El-Shaarawi AH, Piechorsch WW (eds) chapter Maximum entropy sampling. 2nd edn. Wiley, New York, pp 1570–1574
17. Lee J, Williams J (2003) A linear integer programming bound for maximum-entropy sampling. Math Program 94(2-3, Ser. B):247–256
18. Le ND, Zidek JV (2006) Statistical analysis of environmental space-time processes. Springer Series in Statistics. Springer, New York
19. Le N, Zidek J, White R, Cubranic D (2019) EnviroStat: Statistical analysis of environmental space-time processes. Version 0.4-2 https://rdrr.io/cran/EnviroStat/
20. Millard SP (2013) EnvStats: An R package for environmental statistics. Springer, New York
21. NADP (2018) National acidic deposition program, national trends network https://nadp.slh.wisc.edu/ntn/
22. Rencher AC, Christensen WF (2012) Methods of multivariate analysis wiley series in probability and statistics. Wiley, New York
23. Shewry MC, Wynn HP (1987) Maximum entropy sampling. J Appl Stat 14(2):165–170
24. Sebastiani P, Wynn HenryP (2000) Maximum entropy sampling and optimal bayesian experimental design. J R Stat Soc Ser B (Stat Methodol) 62(1):145–157
25. Zidek JV, Sun W, Le ND (2000) Designing and integrating composite networks for monitoring multivariate Gaussian pollution fields. Journal of the Royal Statistical Society. Series C. Applied Statistics 49(1):63–79

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.