Online Resource 03 - Paper Results

How does the temperature vary over time? Evidence on the Stationary and Fractal nature of Temperature Fluctuations

John Dagsvik, Mariachiara Fortuna, Sigmund H. Moen

Affiliations:

John K. Dagsvik, Statistics Norway, Research Department;

Mariachiara Fortuna, freelance statistician, Turin;

Sigmund Hov Moen, Westerdals Oslo School of Arts, Communication and Technology.

Corresponding author:

John K. Dagsvik, E-mail: john.dagsvik@ssb.no

Mariachiara Fortuna, E-mail: mariachiara.fortuna1@gmail.com (reference for code and analysis)

```r
require(tempFGN)
library(knitr)
library(dplyr)
library(tidyr)
library(ggplot2)
```

```r
# DATA PATH
data_final_path <- file.path("data","final")
data_supporting_path <- file.path("data", "supporting")
data_moberg_path <- file.path("data", "moberg")

# OUTPUT PATH
output_supporting_path <- file.path("output", "supporting")
output_table_path <- file.path("output", "table")
output_figure_path <- file.path("output", "figure")
output_temporary_path <- file.path("output", "temporary")
output_manipuated_path <- file.path("output", "manipulated")
```

```r
# ACCESS TO SELECTED TIME SERIES
selected <- read.csv(file.path(data_supporting_path, "T0.SelInfo.csv"), sep=";", dec=",")
country_sel <- selected$Country
station_sel <- selected$Station
njs <- nrow(selected)
data_dir_sel <- file.path(data_final_path, country_sel, paste0(station_sel, ".txt"))
stationame_sel <- paste0(country_sel,", ",station_sel)

# ACCESS TO ALL THE TIME SERIES
all <- read.csv(file.path(data_supporting_path, "T0.TempInfo.csv"), sep=";", dec=",")
country_all <- all$Country
station_all <- all$Station
nja <- nrow(all)
data_dir_all <- file.path(data_final_path, country_all, paste0(station_all, ".txt"))
stationame_all <- paste0(country_all,", ",station_all)

# ACCESS TO MOBERG DATA
moberg <- read.table(file.path(data_moberg_path, "Moberg data.txt"),
                     header = T, na.strings = 99)
Year_m <- moberg[, 1]
Xj_m <- moberg[, 2]
Zj_m <- scale(Xj_m)
Yj_m <- cumsum(Zj_m)
```

Tables

Table 1. Parameter estimation for the selected time series

Estimation results for selected cities based on characteristic function regression and Whittle MLE method. Monthly data.

```r
#====== RUN SEPARATELY, needs about 10 minute to run
#--- 0b. Parameters table
Parameters <- matrix(0, nrow=njs, ncol=6)
colnames(Parameters) <- c("Mu","Sigma","Hc","Hw","SE_Hw","Alpha")
rownames(Parameters) <- paste0(selected$Country,", ",selected$Station)

for (j in 1:njs) {
        #--- 1. Data reading
        data <- read.delim(data_dir_sel[j], header=F, na.strings=99)
        Zm <- monthlyAdj(data, scale=T)$Zm

        #--- 2. Estimation
        Parameters[j,1] <- estim.cf.mu(Zj=Zm) #+ mu_Xj
        Parameters[j,2] <- estim.cf.sigma(Yj=Zm, FBM=F) #* sd_Xj
        Parameters[j,3] <- estim.cf.H(Yj=Zm, FBM=F)
        Parameters[j,4] <- estim.w.H(Zj=Zm)
        Parameters[j,5] <- WhittleEst(Zm)$coefficients[2]
        Parameters[j,6] <- estim.cf.alpha(Yj=Zm, FBM=F)
        }

Parameters <- as.data.frame(Parameters)
write.csv(Parameters, file.path(output_supporting_path,
                                "T1a_selected_parameters_estim.csv"))
```

```r
#=== T1a READING
T1a.selParam <- read.csv(file.path(output_supporting_path,
                                   "T1a_selected_parameters_estim.csv"))
```

```r
#--- 0a. Selected data reading
Hvec <- T1a.selParam$Hc

#--- 0b. Matrix and parameters setting
Boots <- matrix(0, nrow=njs, ncol=4)
colnames(Boots) <- c("SE_Mu", "SE_Sigma", "SE_Hc", "SE_Alpha")
rownames(Boots) <- T1a.selParam[,1]

N <- 1000
Tlenght <- 2000
for (j in 1:njs){
          H <- Hvec[j]
          Parameters <- matrix(0, nrow=N, ncol=4)
          colnames(Parameters) <- c("Mu", "Sigma", "Hc", "Alpha")
          for (i in 1:N){
                         Zjsim <- SimulateFGN(Tlenght, H)
                         Parameters[i,1] <- estim.cf.mu(Zj=Zjsim)
                         Parameters[i,2] <- estim.cf.sigma(Yj=Zjsim, FBM=F)
                         Parameters[i,3] <- estim.cf.H(Yj=Zjsim, FBM=F)
                         Parameters[i,4] <- estim.cf.alpha(Yj=Zjsim, FBM=T)}
                 #--- 3. Table of standard errors
                 SE <- apply(Parameters, 2, sd)
                 Boots[j,] <- SE}
Boots <- as.data.frame(Boots)
write.csv(Boots, file.path(output_supporting_path,
                                "T1b_selected_parameters_bootSE.csv"))
```

```r
#=== T1b READING
T1b.selParamSE <- read.csv(file.path(output_supporting_path,
                                     "T1b_selected_parameters_bootSE.csv"))
```

```r
T1 <- data.frame(
  City = T1b.selParamSE$X,
  Hc = T1a.selParam$Hc,
  SE_Hc = T1b.selParamSE$SE_Hc,
  Hw = T1a.selParam$Hw,
  SE_Hw = T1a.selParam$SE_Hw
  #Alpha = T1a.selParam$Alpha,
  #SE_Alpha = T1b.selParamSE$SE_Alpha
)

write.csv(T1, file.path(output_supporting_path,  "T1_selected_parameters_estimation.csv"))
```

```r
Annual <- matrix(0, nrow=njs, ncol=3)
colnames(Annual) <- c("Hc.a","Hw.a","SE_Hw.a")
rownames(Annual) <- paste0(selected$Country,", ",selected$Station)

for (j in 1:njs) {
        #--- 1. Data reading
        data <- read.delim(data_dir_sel[j], header=F, na.strings=99)
        Xj <- data[,14]
        Zj <- scale(Xj[!is.na(Xj)])
        #--- 2. Estimation
        Annual[j,1] <- estim.cf.H(Yj=Zj, FBM=F)
        Annual[j,2] <- estim.w.H(Zj=Zj)
        Annual[j,3] <- WhittleEst(Zj)$coefficients[2]
        }

Annual <- as.data.frame(Annual)

write.csv(Annual,
          file.path(output_supporting_path, "T1extra_annual_parameters_estim.csv"))
```

```r
T1 <- read.csv(file.path(output_supporting_path,
                         "T1_selected_parameters_estimation.csv"), row.names = 1)

Annual <- read.csv(file.path(output_supporting_path,
                             "T1extra_annual_parameters_estim.csv"), row.names = 1)

T1_ext <- cbind(T1, Annual)

write.csv(T1_ext,
          file.path(output_table_path, "T1ext_selected_parameters_estimation.csv"))
```

```r
kable(T1_ext, digits=3, align='c', escape = T,
      col.names = c("City", "$H_c$", "$SE(H_c)$", "$H_w$", "$SE(H_w)$",
                    "$Ann. H_c$", "$Ann. H_w$", "$Ann. SE(H_w)$"))
```

City	H_c	$SE(H_c)$	H_w	$SE(H_w)$	$Ann. H_c$	$Ann. H_w$	$Ann. SE(H_w)$
Germany, Berlin	0.664	0.009	0.662	0.012	0.726	0.712	0.041
Switzerland, Geneva	0.693	0.001	0.667	0.012	0.845	0.818	0.042

City	H_c	$SE(H_c)$	H_w	$SE(H_w)$	$Ann.H_c$	$Ann.H_w$	$Ann.SE(H_w)$
Switzerland, Basel	0.625	0.011	0.622	0.012	0.664	0.720	0.042
France, Paris	0.733	0.010	0.672	0.012	0.873	0.802	0.042
Sweden, Stockholm	0.681	0.015	0.721	0.012	0.614	0.632	0.041
Italy, Milan	0.724	0.019	0.709	0.012	0.851	0.826	0.043
Czech Republic, Prague	0.684	0.015	0.670	0.012	0.745	0.716	0.043
Hungary, Budapest	0.627	0.011	0.645	0.012	0.682	0.663	0.043
Denmark, Copenhagen	0.755	0.051	0.758	0.013	0.817	0.753	0.045

Table 2. Selected time series and Chi-square test

Chi-square statistics of the FGN hypothesis for selected cities

```
Q <- matrix(0, nrow=njs, ncol=2)
for (j in 1:njs){
 data <- read.delim(data_dir_sel[j], header=F, na.strings=99)
 Zm <- monthlyAdj(data, scale=T)$Zm
 TT <- length(Zm)
 Hc <- estim.cf.H(Yj=Zm, FBM=F)
 Hw <- estim.w.H(Zj=Zm)
 Q[j, ] <- c(Qstat(Zm, H=Hc, TT=TT), Qstat(Zm, H=Hw, TT=TT))
}

monthly_Qt_sel <- data.frame(stationame_sel, Q)

colnames(monthly_Qt_sel) <- c("City", "Q.Hc", "Q.Hw")

#sum(Q>1.96|Q<(-1.96))

write.csv(monthly_Qt_sel, row.names=F,
          file.path(output_table_path, "T2_monthly_Qt_sel.csv"))
```

```
T2.monthlyQtSel <- read.csv(file.path(output_table_path, "T2_monthly_Qt_sel.csv"))
kable(T2.monthlyQtSel, digits=3, align='c',
      col.names = c("City", "$Q(H_c)$", "$Q(H_w)$"))
```

City	$Q(H_c)$	$Q(H_w)$
Germany, Berlin	-0.518	-0.626
Switzerland, Geneva	-0.222	-1.622
Switzerland, Basel	-0.409	-0.484
France, Paris	1.286	-2.791
Sweden, Stockholm	-1.209	1.098
Italy, Milan	-1.339	-2.335
Czech Republic, Prague	-0.198	-0.855
Hungary, Budapest	-0.819	-0.255
Denmark, Copenhagen	-1.006	-0.693

FIGURES

Figure 1. Illustration of statistical self-similarity. Annual temperature for Paris

```r
include_graphics(file.path("output", "manipulated",
                           "F1.Self-similarity yearly monthly data.png"))
```

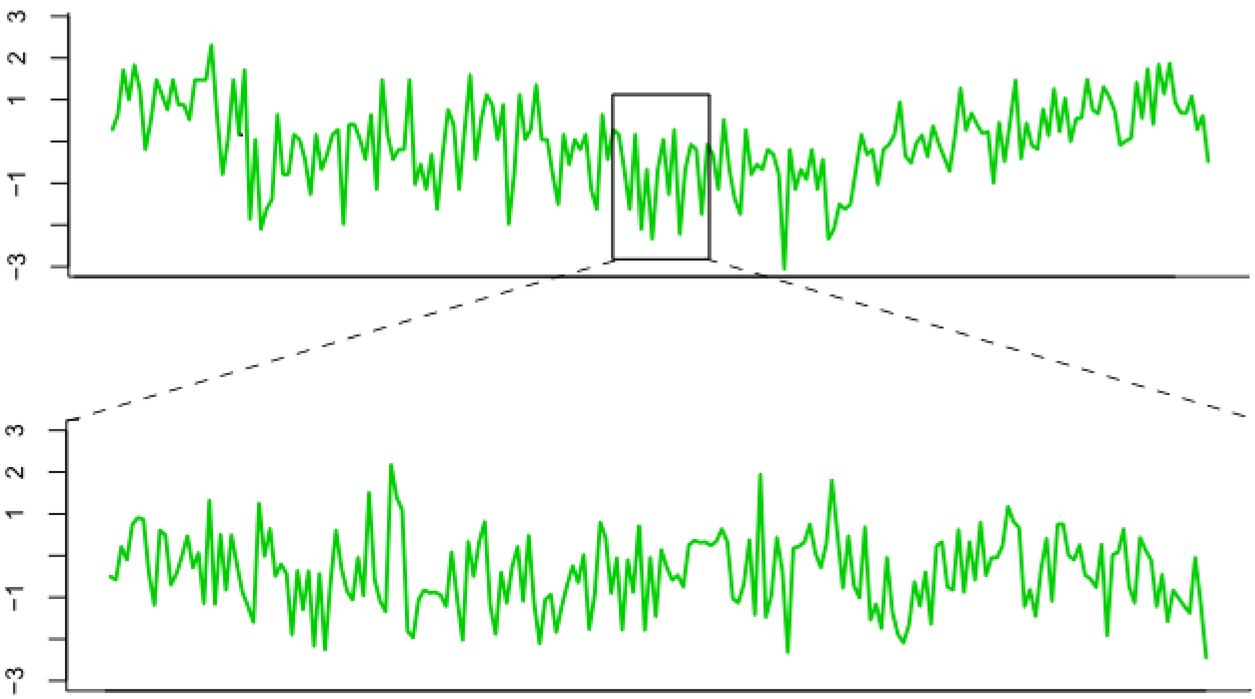
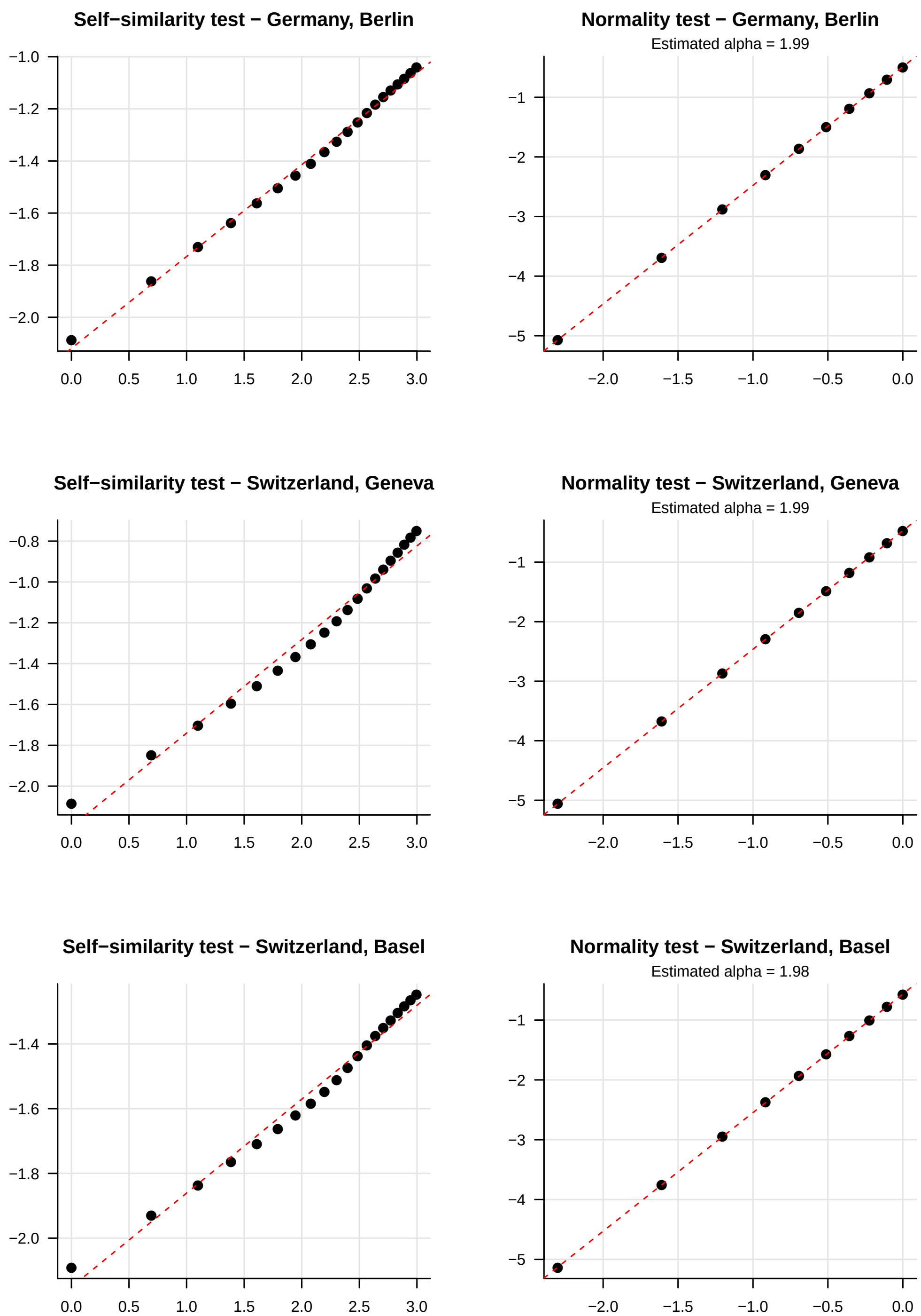


Figure 2. Graphical tests of self-similarity and normality

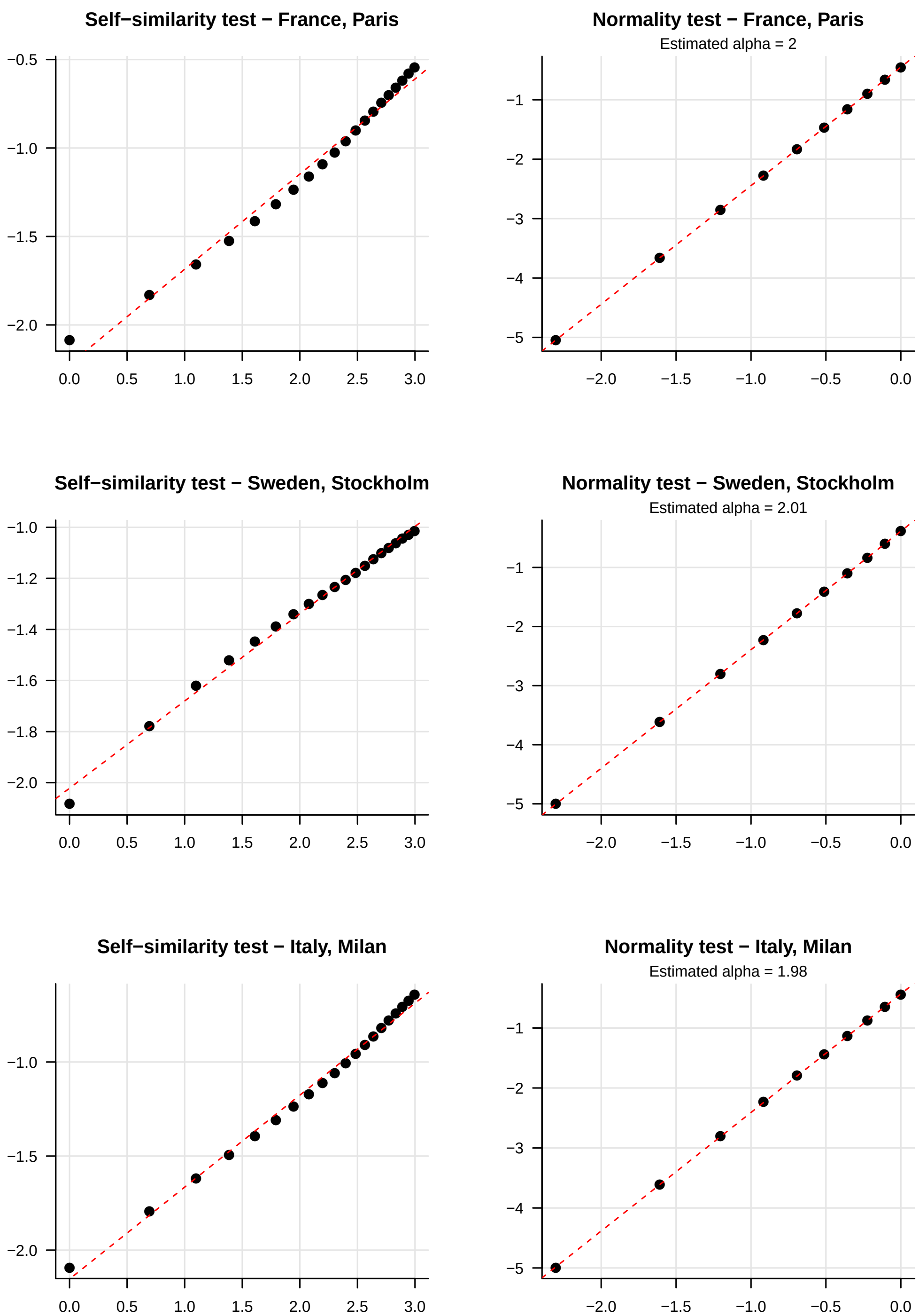
```
pdf(file.path(output_figure_path, "F2_selected_SS_N_test_plot.pdf"),
    paper="a4",width=7, height=10)
par(mfrow=c(3,2))
for (j in 1:njs){
  data <- read.delim(data_dir_sel[j], header=F, na.strings=99)
  Zm <- monthlyAdj(data, scale=T)$Zm
  Ym <- cumsum(Zm)
  fgtSelfSim(Yj=Ym, main=paste("-",stationame_sel[j]), maxd=20,
             cex.dots=1.5, cex.axis=1, cex.main=1.2, subtitle=F)
  fgtNormality(Yj=Ym, main=paste("-",stationame_sel[j]), cex.dots=1.5,
               cex.axis=1, cex.main=1.2)}
dev.off()
```

```
par(mfrow=c(3,2))
for (j in 1:njs){
  data <- read.delim(data_dir_sel[j], header=F, na.strings=99)
  Zm <- monthlyAdj(data, scale=T)$Zm
  Ym <- cumsum(Zm)
  fgtSelfSim(Yj=Ym,main=paste("-",stationame_sel[j]), maxd=20,
             cex.dots=1.5, cex.axis=1, cex.main=1.2, subtitle=F)
  fgtNormality(Yj=Ym,main=paste("-",stationame_sel[j]), cex.dots=1.5,
               cex.axis=1, cex.main=1.2)}
```

```
include_graphics(file.path(output_manipuated_path,
                           "F2_selected_SS_N_test_plot",
                           "p01.pdf"))
```

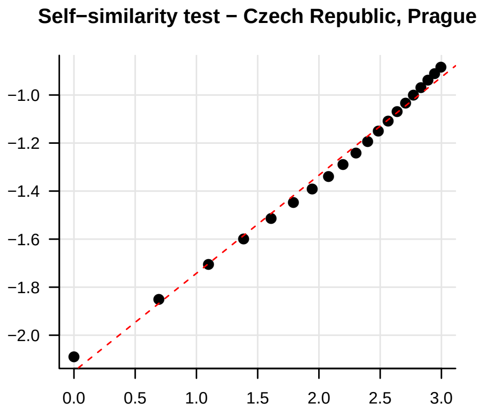

Self−similarity test − Germany, Berlin
Normality test − Germany, Berlin
Estimated alpha = 1.99
Self−similarity test − Switzerland, Geneva
Normality test − Switzerland, Geneva
Estimated alpha = 1.99
Self−similarity test − Switzerland, Basel
Normality test − Switzerland, Basel
Estimated alpha = 1.98

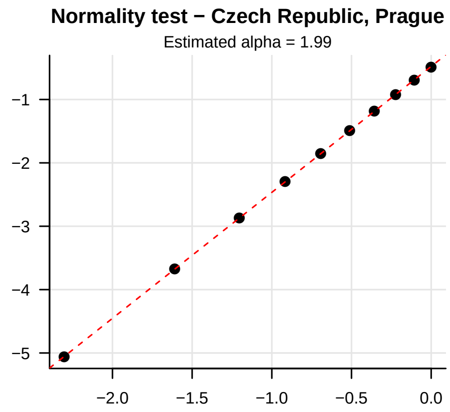
```r
include_graphics(file.path(output_manipuated_path,
                           "F2_selected_SS_N_test_plot",
                           "p02.pdf"))
```

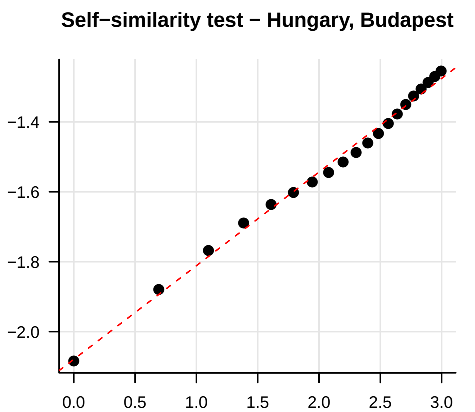


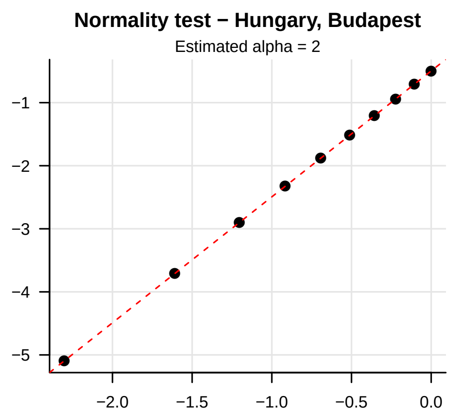
Self−similarity test − France, Paris
Normality test − France, Paris
Estimated alpha = 2
Self−similarity test − Sweden, Stockholm
Normality test − Sweden, Stockholm
Estimated alpha = 2.01
Self−similarity test − Italy, Milan
Normality test − Italy, Milan
Estimated alpha = 1.98

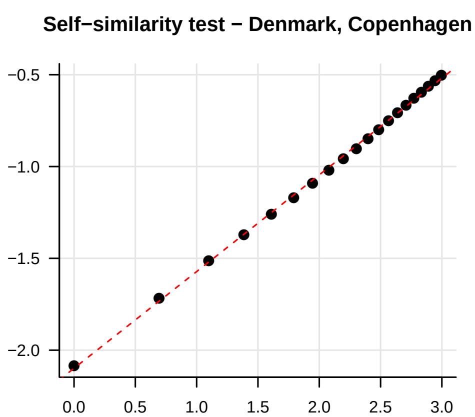
```r
include_graphics(file.path(output_manipuated_path,
                           "F2_selected_SS_N_test_plot",
                           "p03.pdf"))
```


Self−similarity test − Czech Republic, Prague


Normality test − Czech Republic, Prague
Estimated alpha = 1.99


Self−similarity test − Hungary, Budapest


Normality test − Hungary, Budapest
Estimated alpha = 2


Self−similarity test − Denmark, Copenhagen

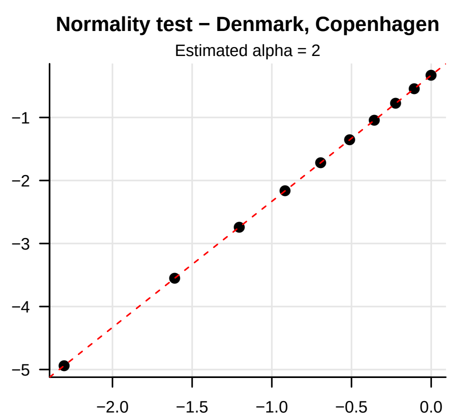
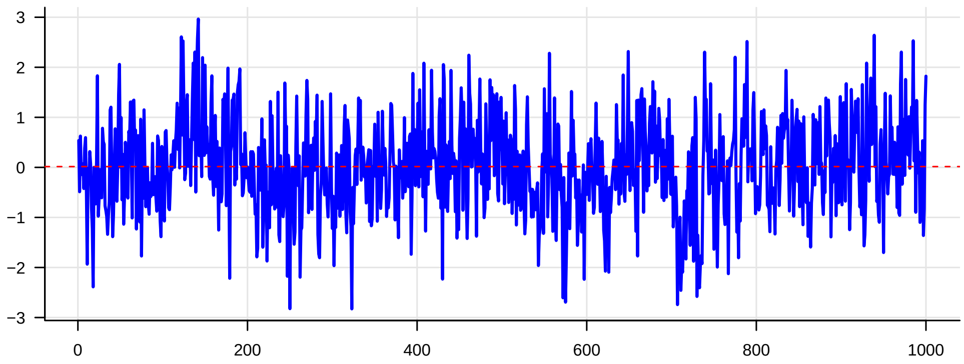

Normality test − Denmark, Copenhagen
Estimated alpha = 2

Figure 3. Simulated FGN process with zero mean and unit variance

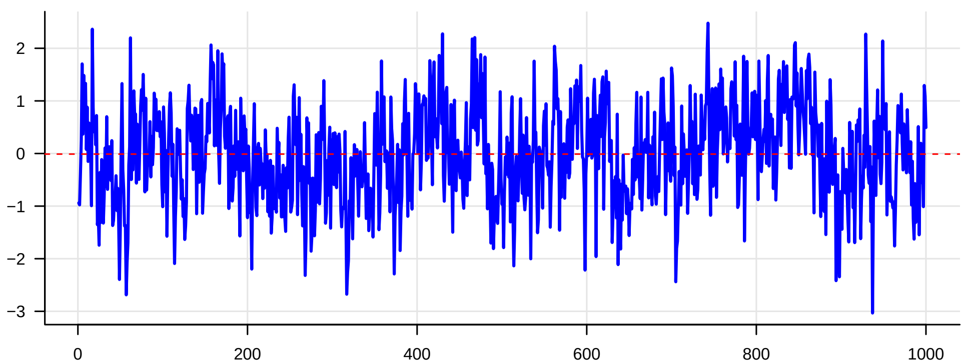
```r
pdf(file.path(output_figure_path, "F3_FGN_simulation.pdf"), paper="a4",
    width=7, height=10)
TT <- 1000
x <- set.seed(123)
par(mfrow=c(3,1))
for (H in c(0.7, 0.8, 0.9)){
  Xj <- simFGN0(TT, H)
  temperaturePlot(Xj = Xj, Year = 1:TT, main = paste0("FGN simulation, H=", H),
                  cex.lab = 1, cex.main = 1.2, cex.axis = 1)
}
dev.off()
```

```r
include_graphics(file.path(output_figure_path, "F3_FGN_simulation.pdf"))
```

FGN simulation, H=0.7



FGN simulation, H=0.8



FGN simulation, H=0.9

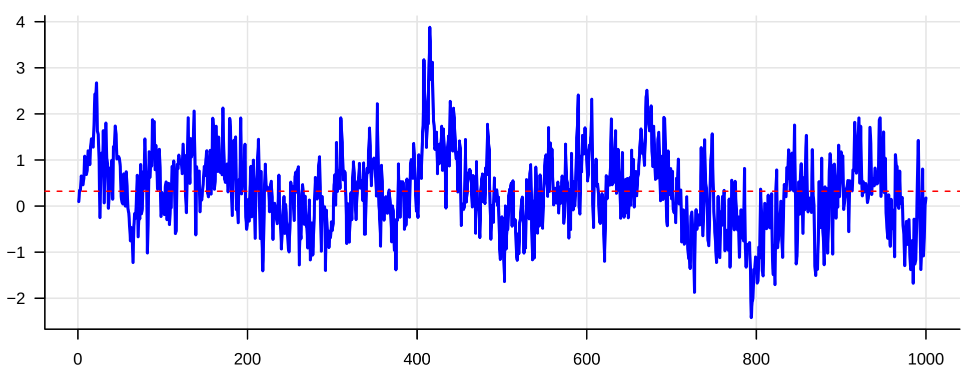


Figure 4. Reconstructed temperature data by Moberg et al. (2005a)

```r
png(file.path(output_figure_path, "F4_moberg_timeseries.png"),
    width = 600, height = 400)
temperaturePlot(Xj = Xj_m, Year = Year_m,
                    cex.lab = 1, cex.main = 1.2, cex.axis = 1)
dev.off()
```

```r
temperaturePlot(Xj = Xj_m, Year = Year_m,
                    cex.lab = 1, cex.main = 1.2, cex.axis = 1)
```

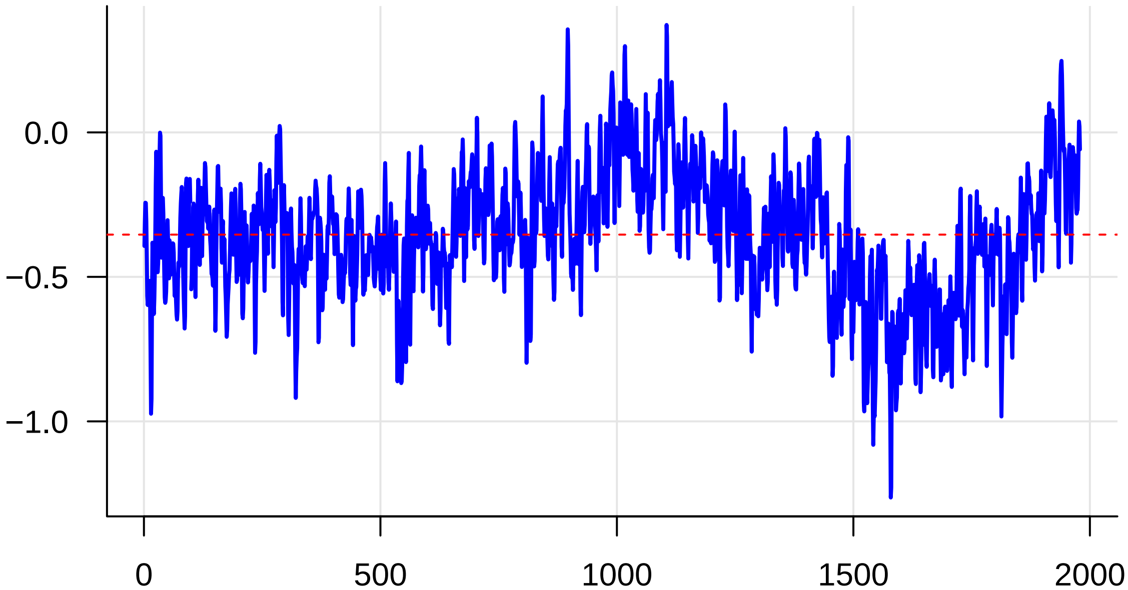
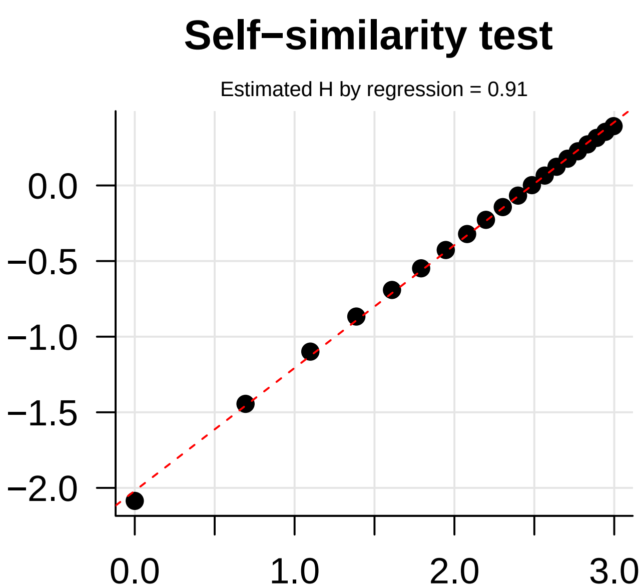


Figure 5. Test of self-similarity and normality

Reconstructed temperature data by Moberg et al. (2005a)

```
pdf(file.path(output_figure_path, "F5a_moberg_SStest.pdf"),
    width = 4, height = 4)
fgtSelfSim(Yj = Yj_m, maxd=20,
           cex.dots=1, cex.axis=1, cex.main=1)$plot
dev.off()
```

```
include_graphics(file.path(output_figure_path, "F5a_moberg_SStest.pdf"))
```




```
pdf(file.path(output_figure_path, "F5b_moberg_Ntest.pdf"),
    width = 4, height = 4)
fgtNormality(Yj = Yj_m, cex.dots=1, cex.axis=1, cex.main=1)$plot
dev.off()
```

```
include_graphics(file.path(output_figure_path, "F5b_moberg_Ntest.pdf"))
```

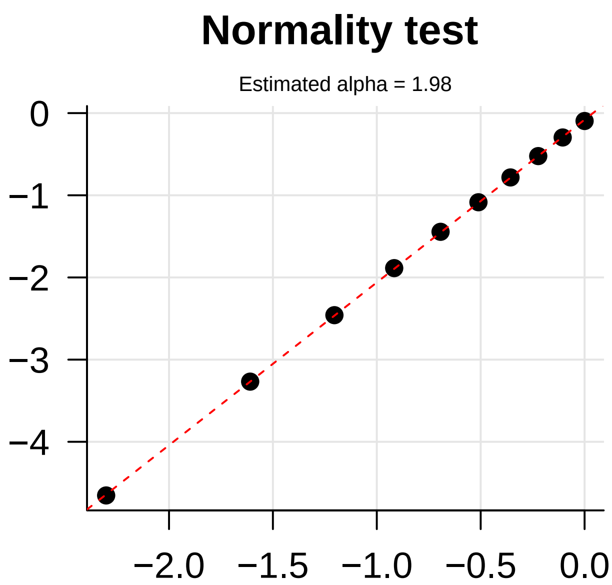

Normality test
Estimated alpha = 1.98
0
−1
−2
−3
−4
−2.0
−1.5
−1.0
−0.5
0.0

Figure 6. Empirical vs FGN autocorrelations

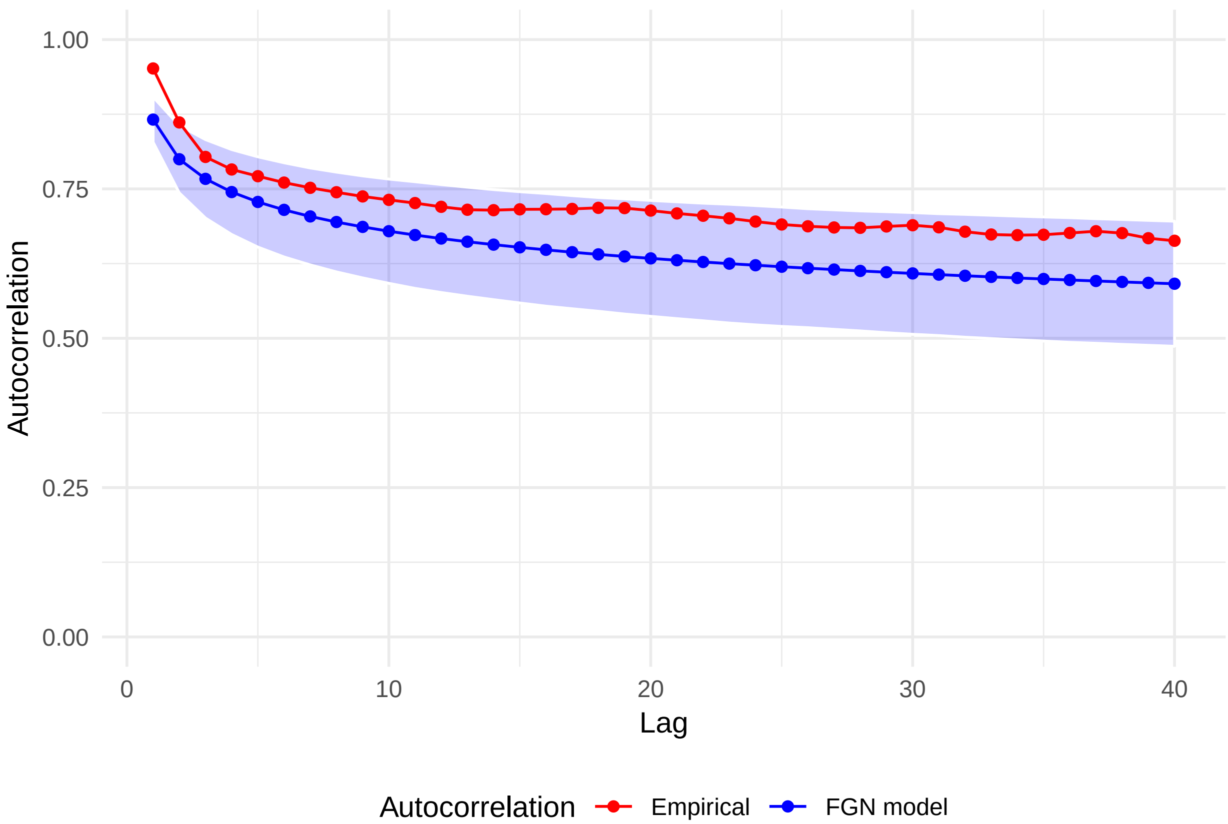
Reconstructed temperature data by Moberg et al. (2005a)

Empirical vs FGN autocorrelation with confidence bands. H=0.95

```
N <- 1000
d <- 40

# Moberg unbiased autocorrelation
unbiased_autocorr_moberg <- unbiased_autocorrelation_function(d, moberg$T)

# Theoretical autocorrelation
gamma_k <- Theoretical_autocorrelation_function(d)

# FGN simulation
fgn_tbl <- sim.multiFGN(N = N, Tj = 2000, H = 0.95)

# Bootstrap estimation
unbiased_autocorr_sim_tbl <- numeric(0)
for (i in 1:length(fgn_tbl)) {
  a <- fgn_tbl %>% select(i)
  X <- as.numeric(unlist(a))
  X_unbiased_autocorr <- as.vector(unbiased_autocorrelation_function(d, X))
  unbiased_autocorr_sim_tbl <- cbind(unbiased_autocorr_sim_tbl,
                                     X_unbiased_autocorr)
}

# Bootstrap mean and sd
unbiased_autocorr_sim_mean <- apply(unbiased_autocorr_sim_tbl, 1, mean)
unbiased_autocorr_sim_sd <- apply(unbiased_autocorr_sim_tbl, 1, sd)

# Autocorrelation tbl
unbiased_autocorr_tbl <-
  data.frame(Theoretical = gamma_k,
             simulated = unbiased_autocorr_sim_mean,
             sd_simulated = unbiased_autocorr_sim_sd,
             Moberg = unbiased_autocorr_moberg) %>%
  mutate(Lag = row_number(),
         CI_inf = Theoretical - 2*sd_simulated,
         CI_sup = Theoretical + 2*sd_simulated)

write.csv(unbiased_autocorr_tbl, file.path(output_supporting_path, "FD4_Unbiased_autocorrelation_tbl.cs
```

```
unbiased_autocorr_tbl <- read.csv(
  file.path(output_supporting_path, "FD4_Unbiased_autocorrelation_tbl.csv"))

unbiased_autocorr_tbl_long <- unbiased_autocorr_tbl %>%
  gather(key = "Autocorrelation", value = "Value",
         -Lag, - simulated, -sd_simulated, -CI_inf, -CI_sup)
```

```r
unbiased_autocorr_tbl_long %>%
ggplot(aes(x = Lag, y = Value, colour = Autocorrelation)) +
ylim(0, 1) +
geom_ribbon(aes(ymin = CI_inf, ymax = CI_sup),
            fill = "blue", col = "white", alpha = 0.2) +
geom_line()  +
geom_point() +
scale_colour_manual(values = c("red", "blue"),
                    labels = c("Empirical", "FGN model")) +
theme_minimal() +
theme(legend.position = "bottom") +
labs(y = "Autocorrelation")
```




```r
# ggsave(file.path(output_figure_path, "F6_Moberg_autocorrelation.png"))
```