Computer Practical 4

Goodness of Fit

Wit, EC., Lomi, A., Lerner, J., Boschi, M. & Lembo, M.

2025-06-04

Remark: Each CP begins by loading data from the previous CP, which is located in the corresponding `_OUTPUT_CP_x_` folder. To ensure this file runs smoothly, the contents of that folder should be copied to the `_INPUT_CP_y_` folder of the current CP.

Session 1: Preparatory Steps

1.1. Installing libraries

```
if (!require("mgcv", quietly = TRUE)) {
  install.packages("mgcv")
  library("mgcv")
} else {
  if (!require("mgcViz", quietly = TRUE)) {
    install.packages("mgcViz")
    library("mgcViz")
  } else {
    if (!require("ggplot2", quietly = TRUE)) {
      install.packages("ggplot2")
      library("ggplot2")
    } else {
      if (!require("survival", quietly = TRUE)) {
        install.packages("survival")
        library("survival")
      } else {
        if (!require("RColorBrewer", quietly = TRUE)){
          install.packages("RColorBrewer")
        } else {
          if (!require("dplyr", quietly = TRUE)){
          install.packages("dplyr")
          } else {
            library("mgcv")
            library("mgcViz")
            library("ggplot2")
            library("survival")
            library("RColorBrewer")
            library(dplyr)
          }
        }
      }
    }
```

```
  }
}
```

Furthermore, we do require functions that allow us to perform GOF in practice. They are contained in the following R file:

```
source("_INPUT_CP_4_/_FUNCTIONS_.R")
```

```
## Warning: il pacchetto 'sde' è stato creato con R versione 4.4.3
##
## Caricamento pacchetto: 'MASS'
## Il seguente oggetto è mascherato da 'package:dplyr':
##
##     select
## Warning: il pacchetto 'fda' è stato creato con R versione 4.4.3
## Warning: il pacchetto 'fds' è stato creato con R versione 4.4.3
## Warning: il pacchetto 'rainbow' è stato creato con R versione 4.4.3
## Warning: il pacchetto 'pcaPP' è stato creato con R versione 4.4.3
## Warning: il pacchetto 'RCurl' è stato creato con R versione 4.4.3
## Warning: il pacchetto 'deSolve' è stato creato con R versione 4.4.3
##
## Caricamento pacchetto: 'fda'
## Il seguente oggetto è mascherato da 'package:graphics':
##
##     matplot
## Warning: il pacchetto 'zoo' è stato creato con R versione 4.4.3
##
## Caricamento pacchetto: 'zoo'
## I seguenti oggetti sono mascherati da 'package:base':
##
##     as.Date, as.Date.numeric
## sde 2.0.18
## Companion package to the book
## 'Simulation and Inference for Stochastic Differential Equations With R Examples'
## Iacus, Springer NY, (2008)
## To check the errata corrige of the book, type vignette("sde.errata")
```

1.2. Loading Data

This Computer Practical (CP) aims to assess the Goodness of Fit (GOF) for some of the models you fitted in previous CPs. We will focus on models fitted using one non-event per event. However, the method can be extended to situations where more than one non-event is used for each event. In such cases — as mentioned in Computer Practical 2 — it is necessary to adjust for the fact that the variance of the score no longer matches the expected value of the second derivative of the score.

We import, for the previous CPs, five models, namely:

- Linear Model from CP1:

```
load("_INPUT_CP_4_/gam_fit.RData")
load("_INPUT_CP_4_/dat_gam_1.RData")
```

```
class(gam_fit)
```

```
## [1] "glm" "lm"
```

```
summary(gam_fit)
```

```
##
## Call:
## glm(formula = y ~ +diff_female + individual_activity + dyadic_activity +
##     closure - 1, family = "binomial", data = dat_gam_1)
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## diff_female         -0.40997    0.14455  -2.836  0.00457 **
## individual_activity  0.06587    0.01370   4.807 1.54e-06 ***
## dyadic_activity      1.69840    0.47198   3.598  0.00032 ***
## closure             -0.09284    0.04165  -2.229  0.02582 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 399.25  on 288  degrees of freedom
## Residual deviance: 149.14  on 284  degrees of freedom
## AIC: 157.14
##
## Number of Fisher Scoring iterations: 10
```

```
gam_fit <- gam(formula = y ~ diff_female + individual_activity + dyadic_activity + closure -
    1, family = "binomial", data = dat_gam_1)
summary(gam_fit)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## y ~ diff_female + individual_activity + dyadic_activity + closure -
##     1
##
## Parametric coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## diff_female         -0.40997    0.14455  -2.836  0.00457 **
## individual_activity  0.06587    0.01370   4.806 1.54e-06 ***
## dyadic_activity      1.69840    0.47200   3.598  0.00032 ***
## closure             -0.09284    0.04165  -2.229  0.02583 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =   -Inf   Deviance explained = -Inf%
```

```
## UBRE = -0.45437  Scale est. = 1         n = 288
```

- Non-linear models from CP2: including time-varying effects, non-linear effects (with our without a log-transform of covariates), and random effects.

```
load("_INPUT_CP_4_/nonlinear_models.RData")
```

We briefly inspect the data again:

```
head(dat_gam_1)
```

```
##   IS_OBSERVED_ev           SOURCE_ev EVENT_INTERVAL individual.activity_ev
## 1              1          |MY|NP|MB|              2                       0
## 2              1          |MY|ME|MB|              3                       2
## 3              1               |MY|              4                       2
## 4              1 |MY|ME|CL|GE|MC|MB|              5                       6
## 5              1          |MY|MB|ME|              6                       9
## 6              1             |ME|MY|              7                       8
##   dyadic.activity_ev closure_ev female_ev diff.female_ev .row_type_ev
## 1                  0          0         1              2           ev
## 2                  1          2         2              2           ev
## 3                  0          0         0              0           ev
## 4                  4          8         3              9           ev
## 5                  7         32         2              2           ev
## 6                  3         12         1              1           ev
##   IS_OBSERVED_nv           SOURCE_nv individual.activity_nv dyadic.activity_nv
## 1              0          |BM|MT|BL|                       0                  0
## 2              0          |JU|EN|TM|                       0                  0
## 3              0               |FV|                       0                  0
## 4              0 |BZ|GA|FV|FE|TM|BS|                       0                  0
## 5              0          |FV|CN|JV|                       0                  0
## 6              0             |BT|FT|                       0                  0
##   closure_nv female_nv diff.female_nv .row_type_nv y female diff_female
## 1          0         1              2          nv 1      0           0
## 2          0         1              2          nv 1      1           0
## 3          0         1              0          nv 1     -1           0
## 4          0         2              8          nv 1      1           1
## 5          0         1              2          nv 1      1           0
## 6          0         1              1          nv 1      0           0
##   individual_activity dyadic_activity closure
## 1                   0               0       0
## 2                   2               1       2
## 3                   2               0       0
## 4                   6               4       8
## 5                   9               7      32
## 6                   8               3      12
```

and review the formulas of the fitted models to recap their differences and the regressors each model includes.

```
gam_fit$formula
```

```
## y ~ diff_female + individual_activity + dyadic_activity + closure -
##     1
```

```
gam_fit_nle$formula
```

```
## y ~ diff_female + dyadic_activity + closure + s(cbind(individual.activity_ev,
##     individual.activity_nv), by = t.mat) - 1
```

```
gam_fit_nle_log$formula
```

```
## y ~ diff_female + dyadic_activity + closure + s(cbind(log(individual.activity_ev +
##     1), log(individual.activity_nv + 1)), by = t.mat) - 1
```

```
gam_fit_tve$formula
```

```
## y ~ diff_female + dyadic_activity + closure + s(EVENT_INTERVAL,
##     by = individual_activity) - 1
```

```
gam_fit_re$formula
```

```
## y ~ s(source_factor, by = t.mat, bs = "re") + diff_female + dyadic_activity +
##     closure + s(EVENT_INTERVAL, by = individual_activity) - 1
```

As discussed in the theoretical part, GOF tests rely on comparing the standardized score to Brownian Bridges.

Since computing Brownian Bridges can be time-consuming for large dimensions, we have precomputed them and stored them in the corresponding object within the input file. Feel free to uncomment the code below if you would like to generate them yourself, or if you need to compute them in a different framework where other dimensions might be required.

```
set.seed(1234)
BB9 <- BB.single(dim.k = 9)
BB10 <- BB.single(dim.k = 10)
```

Session 2: Model Selection vs Goodness of Fit

An important point!

These models were constructed to illustrate the potential of non-linear modeling and to provide interpretations that help attendees understand why a linear model can be restrictive. However, this approach differs from the standard routine, where we would typically first compare competing models and then interpret only the best-fitting model, provided it also passes goodness-of-fit tests.

```
# linear model
AIC(gam_fit)
```

```
## [1] 157.1421
```

```
# model including time-varying effect for individual activity
AIC(gam_fit_tve)
```

```
## [1] 147.8478
```

```
# model including non-linear effect for individual activity
AIC(gam_fit_nle)
```

```
## [1] 154.2472
```

```
# model including non-linear effect for log-individual activity
AIC(gam_fit_nle_log)
```

```
## [1] 153.1042
```

```
# model including random effect for source actor
AIC(gam_fit_re)
```

```
## [1] 182.6415
```

Let us consider the *linear model as the baseline*. Among the fitted models, the one *including a time-varying effect* for individual activity achieves the *lowest AIC*, indicating the best fit while maintaining a good balance between fit and complexity. The models incorporating a non-linear effect for individual activity or its logarithm perform only better than the linear model. The *worst-performing model* is the one *including a random effect* for the source actor, indicating that this addition increases complexity without providing a corresponding improvement in fit.

Among this set of models, we would select the one *including a time-varying effect for individual activity*, as it provides the best balance between model fit and complexity.

Session 3: Goodness of Fit Evaluation

In the following, we will assess the goodness of fit for all five models.

3.1. Linear Model

```
GOF_linear_DF = GOF_univariate(gam.fit = gam_fit,
                               index = 1)
GOF_linear_DF[[1]]
```

```
## [1] 0.05216391
```

```
GOF_linear_IA = GOF_univariate(gam.fit = gam_fit,
                               index = 2)
GOF_linear_IA[[1]]
```

```
## [1] 0.2183708
```

```
GOF_linear_DA = GOF_univariate(gam.fit = gam_fit,
                               index = 3)
GOF_linear_DA[[1]]
```

```
## [1] 0.8858255
```

```
GOF_linear_C = GOF_univariate(gam.fit = gam_fit,
                              index = 4)
GOF_linear_C[[1]]
```

```
## [1] 0.3187251
```

As you can see, the null hypothesis of model adequacy would not be rejected. However, based on model selection criteria, we would not have chosen this model as the best in terms of predictive performance.

3.2. Models with non-linear effects for individual activity

First of all, we need to identify what each column of the model matrix represents. We can do that by inspecting the coefficients of the fitted model.

```
coefficients(gam_fit_nle)[1]
```

```
## diff_female
##  -0.3603861
```

```
coefficients(gam_fit_nle)[2]
```

```
## dyadic_activity
##        1.608857
```

```r
coefficients(gam_fit_nle)[3]
```

```
##     closure
## -0.09159725
```

```r
coefficients(gam_fit_nle)[4:12]
```

```
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.1
##                                                      1.20578080
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.2
##                                                     -1.93890493
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.3
##                                                     -0.54685905
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.4
##                                                      0.84106206
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.5
##                                                      0.03322874
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.6
##                                                     -0.66833616
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.7
##                                                     -0.30142198
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.8
##                                                      2.76270352
## s(cbind(individual.activity_ev, individual.activity_nv)):t.mat.9
##                                                      3.28241323
```

```r
GOF_nle_DF = GOF_univariate(gam.fit = gam_fit_nle,
                            index = 1)
GOF_nle_DF[[1]]
```

```
## [1] 0.04875907
```

```r
GOF_nle_DA = GOF_univariate(gam.fit = gam_fit_nle,
                            index = 2)
GOF_nle_DA[[1]]
```

```
## [1] 0.8698286
```

```r
GOF_nle_C = GOF_univariate(gam.fit = gam_fit_nle,
                           index = 3)
GOF_nle_C[[1]]
```

```
## [1] 0.2626076
```

```r
GOF_nle_IA = GOF_multivariate(gam.fit = gam_fit_nle,
                  index = 4:12, BB.stat = BB9[[2]])
GOF_nle_IA[[1]]
```

```
## [1] 0.0164
```

```r
GOF_nle_log_DF = GOF_univariate(gam.fit = gam_fit_nle_log,
                            index = 1)
GOF_nle_log_DF[[1]]
```

```
## [1] 0.05054661
```

```r
GOF_nle_log_DA = GOF_univariate(gam.fit = gam_fit_nle_log,
                            index = 2)
GOF_nle_log_DA[[1]]
```

```
## [1] 0.9420363
```

```
GOF_nle_log_C = GOF_univariate(gam.fit = gam_fit_nle_log,
                               index = 3)
GOF_nle_log_C[[1]]
```

```
## [1] 0.1582014
```

```
GOF_nle_log_IA = GOF_multivariate(gam.fit = gam_fit_nle_log,
                  index = 4:12, BB.stat = BB9[[2]])
GOF_nle_log_IA[[1]]
```

```
## [1] 0.0192
```

Both non-linear effect of log-individual activity and non-linear effect of individual activity (without log transformation) are rejected.

3.3. Models with time-varying effects for individual activity

This is the model we would be asked to interpret based on the earlier model selection. All included effects are deemed adequate.

```
coefficients(gam_fit_tve)[1]
```

```
## diff_female
##  -0.3133983
```

```
coefficients(gam_fit_tve)[2]
```

```
## dyadic_activity
##        1.571009
```

```
coefficients(gam_fit_tve)[3]
```

```
##     closure
## -0.08280746
```

```
coefficients(gam_fit_tve)[4:13]
```

```
##  s(EVENT_INTERVAL):individual_activity.1
##                               -0.45940589
##  s(EVENT_INTERVAL):individual_activity.2
##                                0.21219721
##  s(EVENT_INTERVAL):individual_activity.3
##                               -0.15391632
##  s(EVENT_INTERVAL):individual_activity.4
##                               -0.01617049
##  s(EVENT_INTERVAL):individual_activity.5
##                                0.05471957
##  s(EVENT_INTERVAL):individual_activity.6
##                               -0.01791173
##  s(EVENT_INTERVAL):individual_activity.7
##                                0.05814590
##  s(EVENT_INTERVAL):individual_activity.8
##                                0.03488562
##  s(EVENT_INTERVAL):individual_activity.9
##                               -0.26118123
```

```
## s(EVENT_INTERVAL):individual_activity.10
##                                -0.26315025
```

```
GOF_tve_DF = GOF_univariate(gam.fit = gam_fit_tve,
                            index = 1)
GOF_tve_DF[[1]]
```

```
## [1] 0.06900206
```

```
GOF_tve_DA = GOF_univariate(gam.fit = gam_fit_tve,
                            index = 2)
GOF_tve_DA[[1]]
```

```
## [1] 0.8974657
```

```
GOF_tve_C = GOF_univariate(gam.fit = gam_fit_tve,
                            index = 3)
GOF_tve_C[[1]]
```

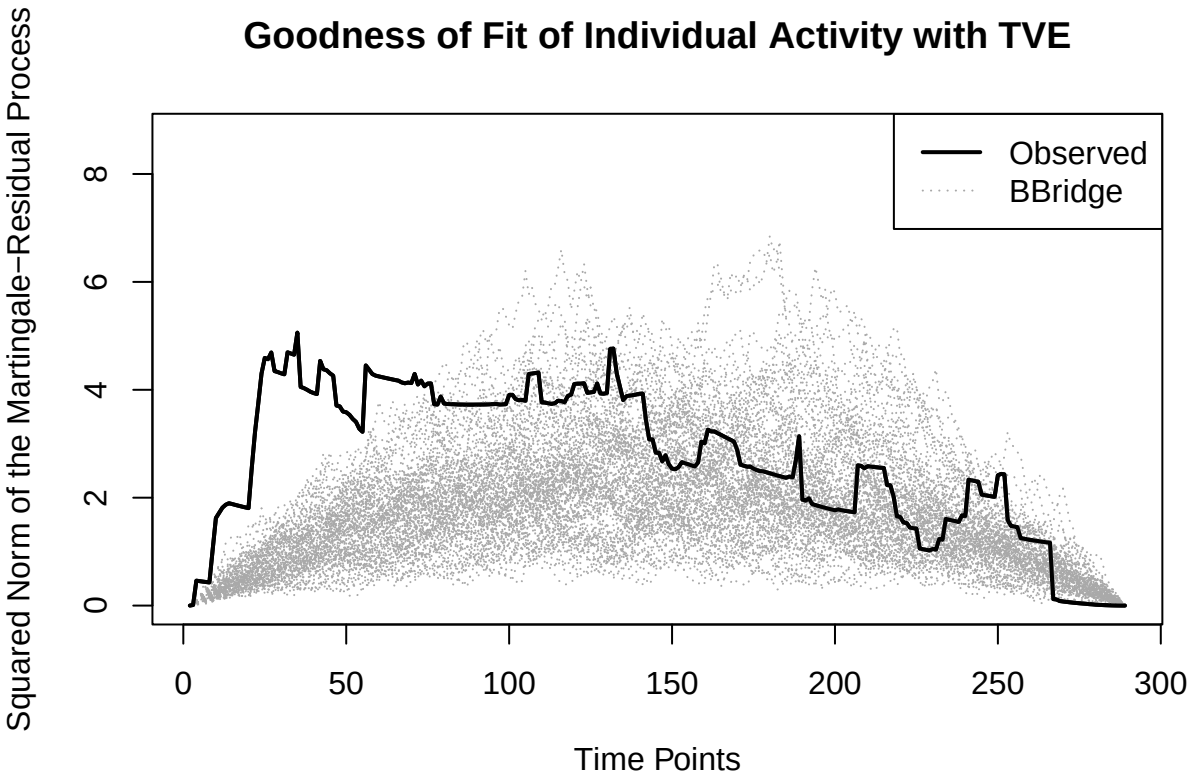
```
## [1] 0.2282975
```

```
GOF_tve_IA = GOF_multivariate(gam.fit = gam_fit_tve,
                  index = 4:13, BB.stat = BB10[[2]])
GOF_tve_IA[[1]]
```

```
## [1] 0.1534
```

We also include a graphical representation of the Martingale process:

```
plot(
  dat_gam_1$EVENT_INTERVAL,
  apply(GOF_tve_IA[[2]], 1, function(x) crossprod(x,x)),
  type="l",
  lwd=2,
  col="black",
  xlab="Time Points",
  ylab="Squared Norm of the Martingale-Residual Process",
  main="Goodness of Fit of Individual Activity with TVE",
  ylim=c(0,max(BB10[[1]]))
)
for (iter in 1:100){
  lines(
    dat_gam_1$EVENT_INTERVAL,
    BB10[[1]][seq(1, 2000,
                  length.out=length(dat_gam_1$EVENT_INTERVAL)),iter],
    col="darkgray", lwd=1,
    lty=3)
}
lines(
  dat_gam_1$EVENT_INTERVAL,
  apply(GOF_tve_IA[[2]], 1, function(x) crossprod(x,x)),
  lwd=2,
  ylim= c(0,10),
  col="black")
legend("topright",c("Observed", "BBridge"),
       lty = c(1,3),lwd = c(2,1), col = c("black","darkgray"))
```


Goodness of Fit of Individual Activity with TVE
Observed
BBridge
Squared Norm of the Martingale−Residual Process
0
2
4
6
8
0
50
100
150
200
250
300
Time Points