GPoM : 2 PreProcessing

Sylvain Mangiarotti & Mireille Huc

2018-07-26

Pre-processing for global modelling

To obtain Ordinary Differential Equations (ODEs) from time series requires the computation of its derivative(s). A careful preprocessing of the observed time series may thus be necessary before starting the modelling process. Although derivatives will be computed automatically when using the `gPoMo` function, some preprocessing may generally be necessary. This vignette aims to show simple examples of time series preprocessing for global modelling application. In practice, various types of preprocessing may be necessary that will depend on the quality of the data set (time sampling and time series length, level of noise, presence of gaps, etc.). In the present vignette, we will mainly focus on the problem of subsampled time series. We will illustrate the usefulness to resample the data (with the aim to improve the quality of the derivatives) and some of the limitations that may be found, especially when considering derivatives of higher degree.

The required preprocessing may also depend on other factors such as: the dimension of underlying dynamics, the number of observed time series, the degree of observability, etc.

Single time series

When only a single time series is observed, a global model of canonical form may be expected:

$dX_1/dt = X_2$

$dX_2/dt = X_3$

...

$dX_n/dt = F(X_1, X_2, ..., X_n)$,

with X_1 the observed variable, and X_2, X_3, etc. its successive derivatives. The GPoM package is based on polynomial formulations. For the canonical formulation, its aim is to obtain a polynomial approximation of the function F. An approximation of F can be obtained using the `gPoMo` function (see vignette `3 Modelling`). In practice, the canonical form will require the computations of n derivatives in order to have the following $n+1$ time series: $(X_1, X_2, ..., X_n, dX_n/dt)$.

Chaotic systems are well adapted to illustrate the performances of the global modelling approach because they are not trivial study cases: they are nonlinear and their behavior is unpredictable at long term [1]. The Rössler system introduced by Otto Rössler in 1976 is used here[2] to illustrate the performances of the tool. This system is defined as:

$dx/dt = -y - z$

$dy/dt = x + ay$

$dz/dt = b + z(x - c)$.

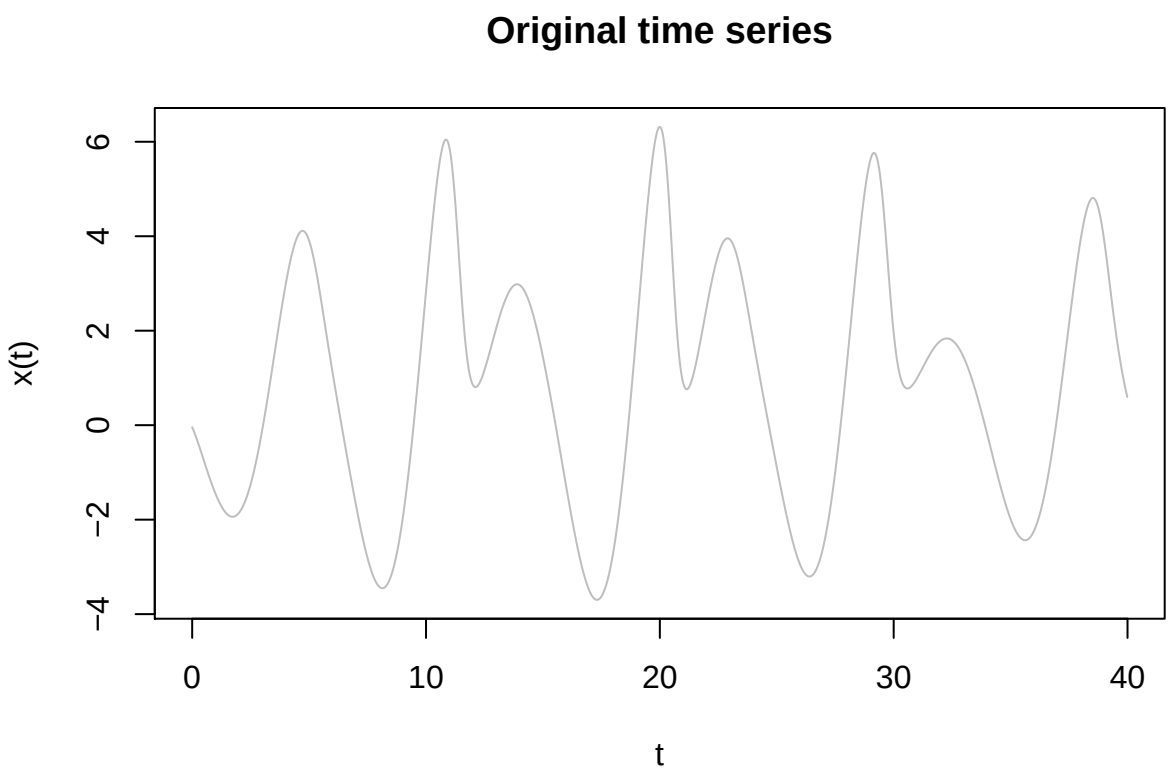
One of the interests of this system is to have different degrees of observability depending on the considered variable. This degree of observability results from the system nonlinearities. The question of observability is of first importance for global modelling since the model reconstruction may be either easy, hard or even

[1] S. Mangiarotti, Modélisation globale et caractérisation topologique de dynamiques environnementales: de l'analyse des enveloppes fluides et du couvert de surface de la Terre à la caractérisation topolodynamique du chaos, Habilitation to Direct Researches, Université de Toulouse 3, 2014.

[2] O. Rössler, 1976. An Equation for Continuous Chaos, *Physics Letters*, **57A**(5), p. 397-398.

completely impossible depending on the degree of observability[3]. Time series of such equations can be easily generated with the GPoM package (see vignette `1 Conventions`). A ready-to-use data set of the Rössler-1976 system that can be directly loaded is also provided in the package:

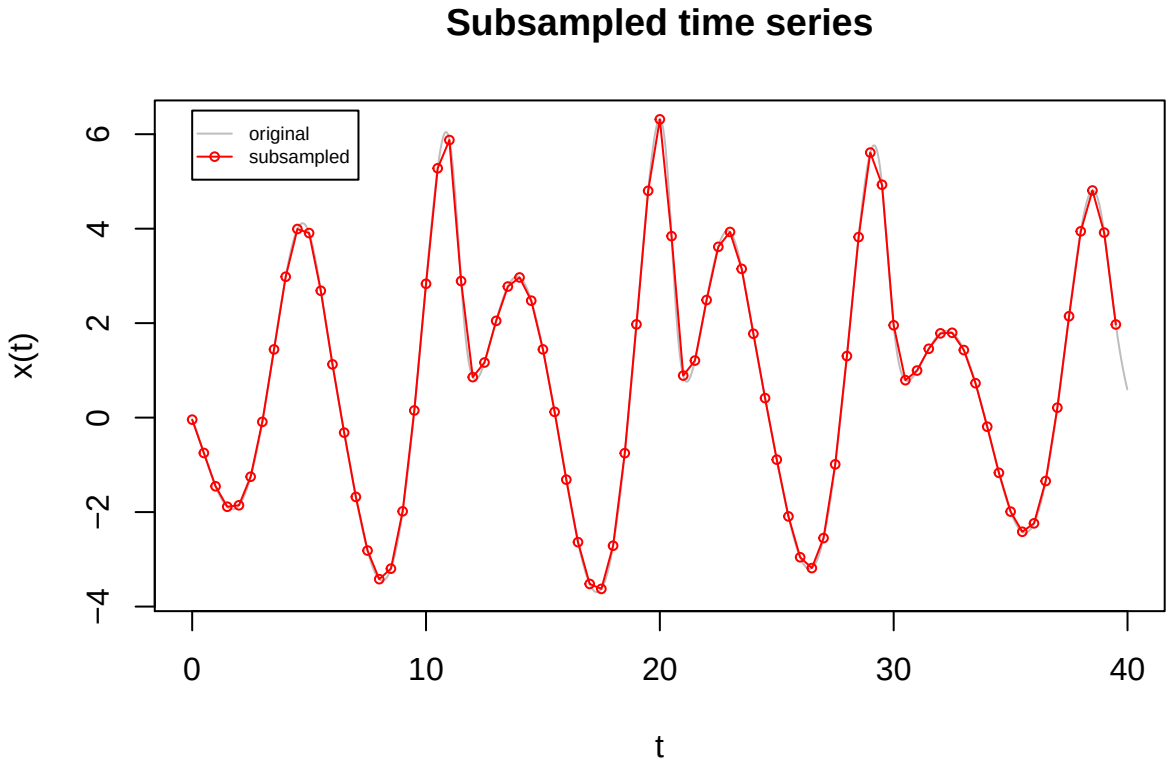
```
# load data
data("Ross76")
# plot
tin <- Ross76[,1]
data <- Ross76[,2:4]
plot(tin, data[,1], xlab = 't', ylab = 'x(t)', main = 'Original time series',
     type='l', col = 'gray')
```



The time series $x(t)$ plotted above shows an irregular behavior and will be used hereafter to illustrate the global modelling with a canonical formulation. To illustrate the problem of time sampling, the time series is subsampled as follows:

```
plot(tin, data[,1], xlab = 't', ylab = 'x(t)',
     main = 'Subsampled time series', type='l', col = 'gray')
# subsampling:
Ross76us <- Ross76[seq(1,4000,by=50),]
# plot
tus <- Ross76us[,1]
datus <- Ross76us[,2:4]
lines(tus, datus[,1], type='p', col='red', cex = 0.6)
lines(tus, datus[,1], type='l', col='red')
legend(0,6.5,c("original", "subsampled"), col=c('gray', 'red'),
       lty=c(1,1), pch=c(NA,1), cex=0.6)
```

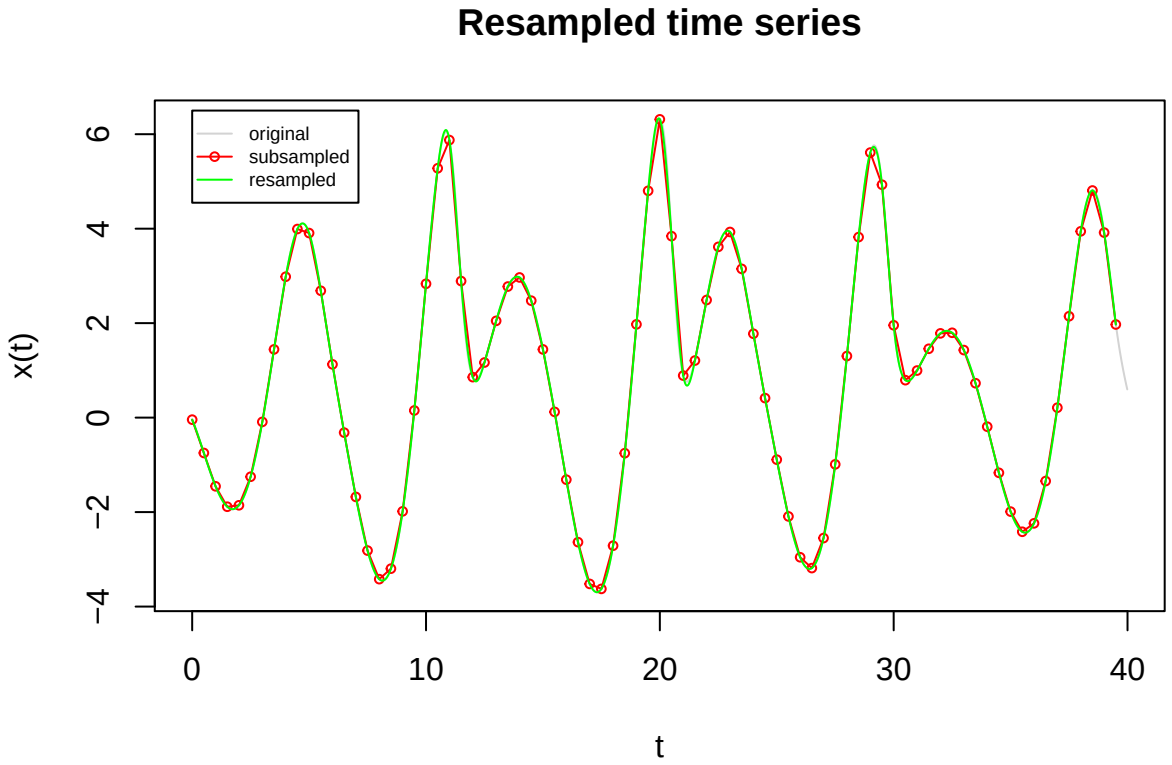
[3]C. Letellier, L. A. Aguirre, & J. Maquet, 2005. Relation between observability and differential embeddings for nonlinear dynamics, *Physical Review E*, **71**(6), 066213.



The subsampling leads to loose some information. However, the general shape is obviously kept.

A spline can be used to resample the signal at the original sampling rate.

```
plot(tin, data[,1], xlab = 't', ylab = 'x(t)',
     main = 'Resampled time series', type='l', col = 'lightgray')
# subsampling:
Ross76us <- Ross76[seq(1,4000,by=50),]
# plot
tus <- Ross76us[,1]
datus <- Ross76us[,2:4]
lines(tus, datus[,1], type='p', col='red', cex = 0.6)
lines(tus, datus[,1], type='l', col='red')
#
xout <- seq(min(tus), max(tus), by = 0.01)
rspl <- spline(tus, y = datus[,1], method = "fmm", xout = xout)
# plot resampled
lines(rspl$x, rspl$y, type='l', col='green')
legend(0,6.5,c("original", "subsampled", "resampled"),
       col=c('lightgray', 'red', 'green'), lty=c(1,1,1),
       pch=c(NA,1, NA), cex=0.6)
```



In the present case, the spline resampling appears quite efficient to retrieve the original signal. Note that alternative methods may be preferred (depending on the dynamical behavior considered, on the rate of subsampling, on the signal quality, etc.)

The `drvSucc` function of the GPoM package aims to estimate temporal derivatives from a time series. It is based on the algorithm introduced by Savitzky & Golay in 1964[4]. This function is here applied to the three time series $x(t)$ presented upper: (1) the original Rössler time series, (2) its subsampling, and (3) its resampling to the original sampling using splines.

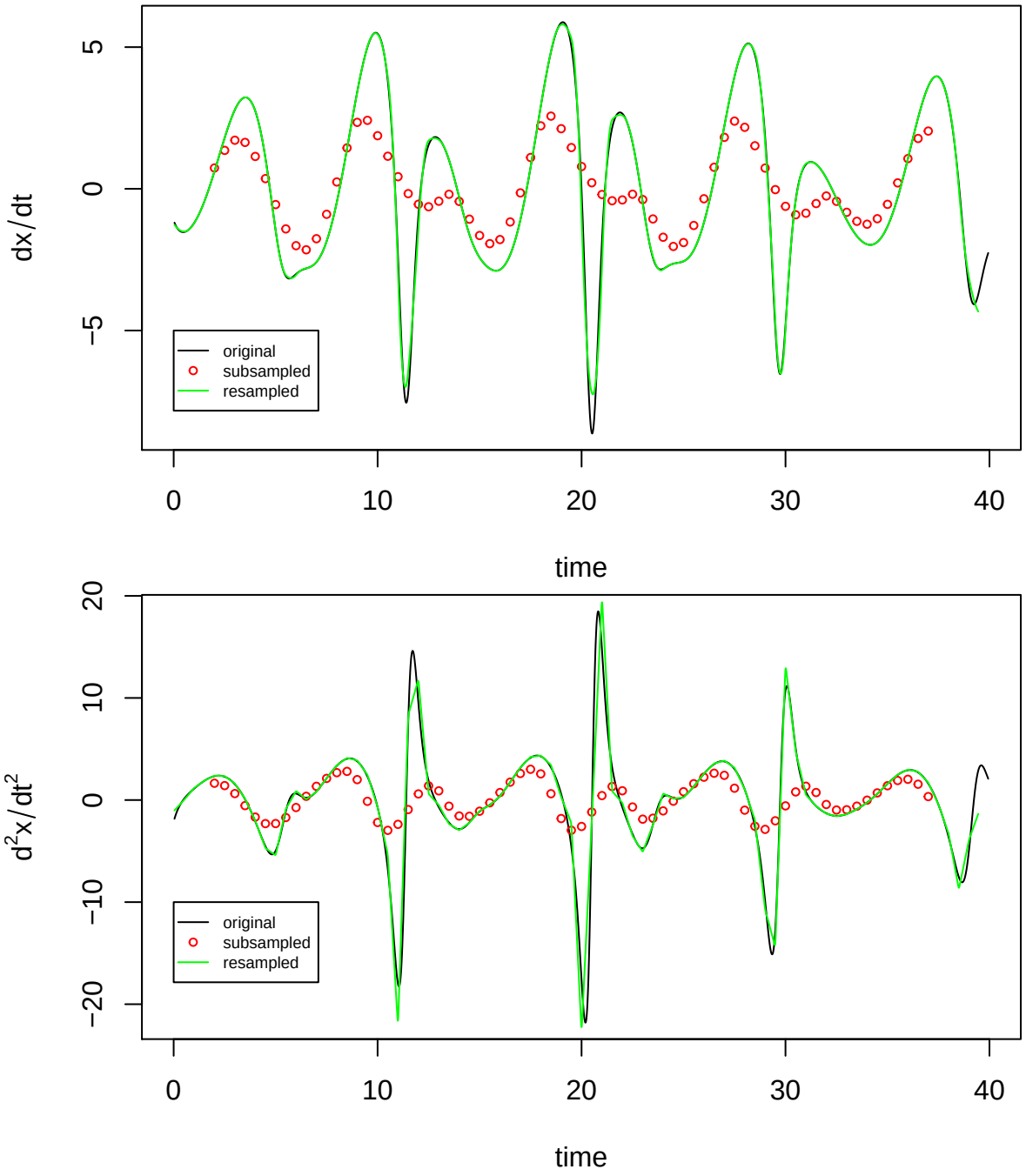
```
# from original signal
drv1 <- drvSucc(tin, data[,1], nDeriv=2)
# from subsampled signal
drv2 <- drvSucc(tus, datus[,1], nDeriv=2)
# from resampled signal
drv3 <- drvSucc(rspl$x, rspl$y, nDeriv=2)
```

The obtained time series of the first and second derivatives are plotted hereafter. Obviously, much of the original signal (in black) is lost when computing the derivatives from the subsampled time series (in red). Contrarily, much of the original signal is retrieved when estimated from the resampled signal (in green), which clearly illustates the usefulness to resample the signal with appropriate algorithms before computing the derivatives.

```
# plot resulting output as a function of time
# first derivative
plot(drv1$tout, drv1$seriesDeriv[,2], type='l', xlab = 'time',
     ylab= expression(dx/dt))
lines(drv2$tout, drv2$seriesDeriv[,2], type='p', cex = 0.6, col = 'red')
lines(drv3$tout, drv3$seriesDeriv[,2], type='l', col = 'green')
legend(0,-5,c("original", "subsampled", "resampled"),
```

[4] A. Savitzky & M. J. Golay, 1964. Smoothing and differentiation of data by simplified least squares procedures, *Analytical Chemistry*, **36**(8), 1627-1639.

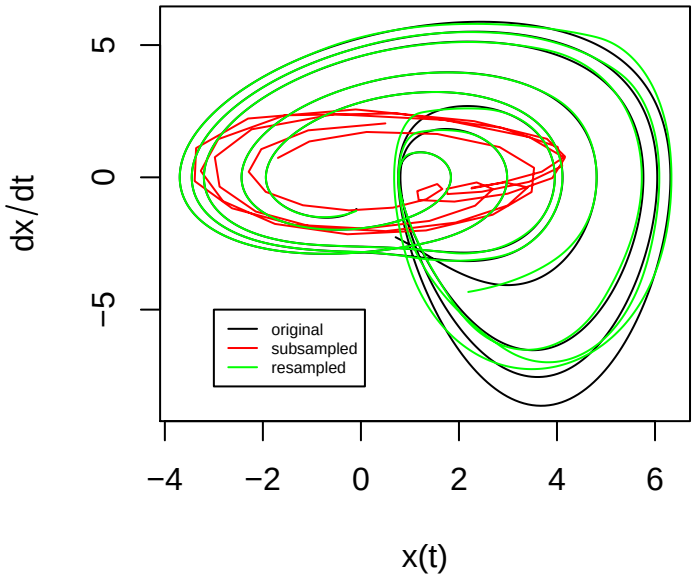
```
       col=c('black', 'red', 'green'), lty=c(1,NA,1), pch=c(NA,1, NA), cex=0.6)
# second derivative
plot(drv1$tout, drv1$seriesDeriv[,3], type='l', xlab = 'time',
     ylab= expression(d^2*x/dt^2))
lines(drv2$tout, drv2$seriesDeriv[,3], type='p', cex = 0.6, col = 'red')
lines(drv3$tout, drv3$seriesDeriv[,3], type='l', col = 'green')
legend(0,-10,c("original", "subsampled", "resampled"),
       col=c('black', 'red', 'green'), lty=c(1,NA,1), pch=c(NA,1, NA), cex=0.6)
```

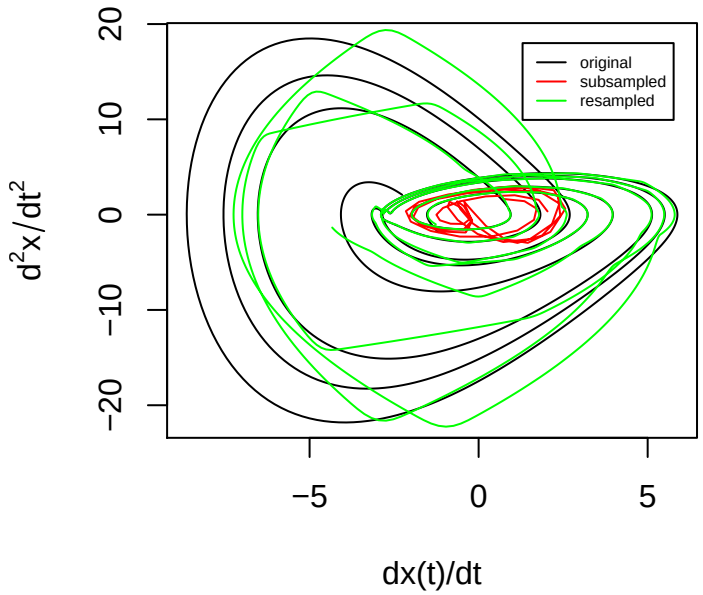


Phase portraits can also be compared. Two projections are shown hereafter: $(x, dx/dt)$ and $(x, d^2x/dt^2)$. Obviously, quite much of the information of the original phase portraits (plotted in black) is lost when

the phase portrait is reconstructed from the subsampled time series (in red). Contrarily, the projection $(x, dx/dt)$ of the phase portrait is retrieved almost perfectly when reconstructed from the resampled time series (in green). The second phase portrait $(x, d^2x/dt^2)$ shows that some drawbacks were introduced by the resampling, although this processing clearly improved the signal compared to the subsampled reconstruction.

```
# output variable (smoothed) and its derivatives
plot(drv1$seriesDeriv[,1], drv1$seriesDeriv[,2], type='l', xlab = 'x(t)',
     ylab= expression(dx/dt))
lines(drv2$seriesDeriv[,1], drv2$seriesDeriv[,2], type='l', col = 'red')
lines(drv3$seriesDeriv[,1], drv3$seriesDeriv[,2], type='l', col = 'green')
legend(-3,-5,c("original", "subsampled", "resampled"),
       col=c('black', 'red', 'green'), lty=1, cex=0.5)
# second derivative
plot(drv1$seriesDeriv[,2], drv1$seriesDeriv[,3], type='l', xlab = 'dx(t)/dt',
     ylab= expression(d^2*x/dt^2))
lines(drv2$seriesDeriv[,2], drv2$seriesDeriv[,3], type='l', col = 'red')
lines(drv3$seriesDeriv[,2], drv3$seriesDeriv[,3], type='l', col = 'green')
legend(1.3,18,c("original", "subsampled", "resampled"),
       col=c('black', 'red', 'green'), lty=1, cex=0.5)
```





This preprocessing illustrates (1) the usefulness to have a proper sampling of the observed process; (2) the usefulness to resample the observed time series at a better time sampling to improve the estimates of the derivatives; (3) some of the limitations reached for the derivatives of higher degree. Subsampling will have a direct and drastic impact when applying the global modelling technique. Resampling can be very efficient to retrieve the original signal and to obtain a good model[5]

Since the derivatives will be computed automatically when using the `gPoMo` function, it is important to keep in mind that their quality will directly depend on the characteristic of the time series used as input.

Multiple time series

When several time series are available, global models of the following form can be expected:

$dX_1/dt = f_1(X_1, X_2, ..., X_n)$

$dX_2/dt = f_2(X_1, X_2, ..., X_n)$

...

$dX_n/dt = f_n(X_1, X_2, ..., X_n).$

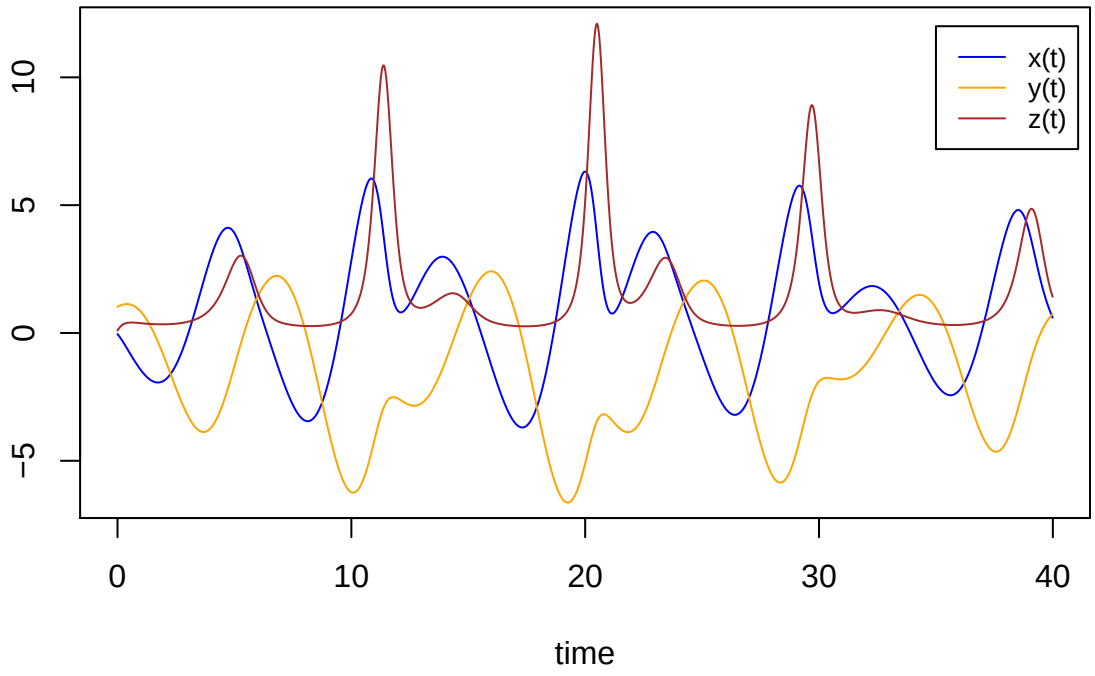
where the X_i, are the observed variables. The GPoM package can be used to obtain a polynomial approximation of the functions f_i (see vignette `3 Modelling`). This formulation will thus require the computations of the first derivatives of each time series in order to have $(X_1, X_2, ..., X_n, dX_1/dt, dX_2/dt, ...dX_n/dt)$.

Here again the Rössler system (1976) is used to illustrate the performances of the global modelling approach.

```
# load
data("Ross76")
# plot
tin <- Ross76[,1]
data <- Ross76[,2:4]
plot(tin, data[,1], ylim = c(-6.5,12), xlab = 'time', ylab = '', type='l', col='blue')
lines(tin, data[,2], type='l',  col='orange')
```

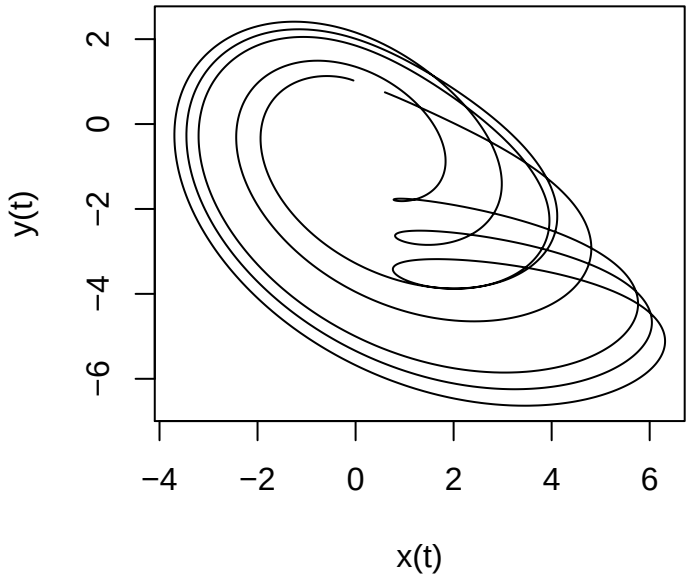
[5]S. Mangiarotti, 2018. The global modelling classification technique applied to the detection of chaotic attractors. *Supplementary Material A* to "Can the global modelling technique be used for crop classification?" by S. Mangiarotti, A.K. Sharma, S. Corgne, L. Hubert-Moy, L. Ruiz, M. Sekhar, Y. Kerr, 2018. *Chaos, Solitons & Fractals*, **106**, 363-378.

```
lines(tin, data[,3], type='l',  col='brown')
legend(35,12,c("x(t)", "y(t)", "z(t)"), col=c('blue', 'orange', 'brown'), lty=1, cex=0.8)
```

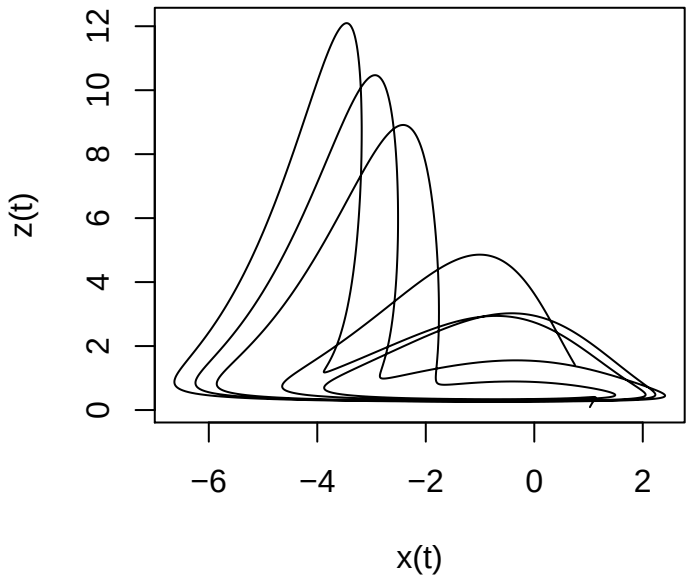


The phase space of the original Rössler system can be reconstructed for the following set of time series $x(t)$, $y(t)$ and $z(t)$: The following reconstructions correspond to projections (x, y) and (x, z):

```
plot(data[,1], data[,2], type='l', xlab = 'x(t)', ylab = 'y(t)')
```



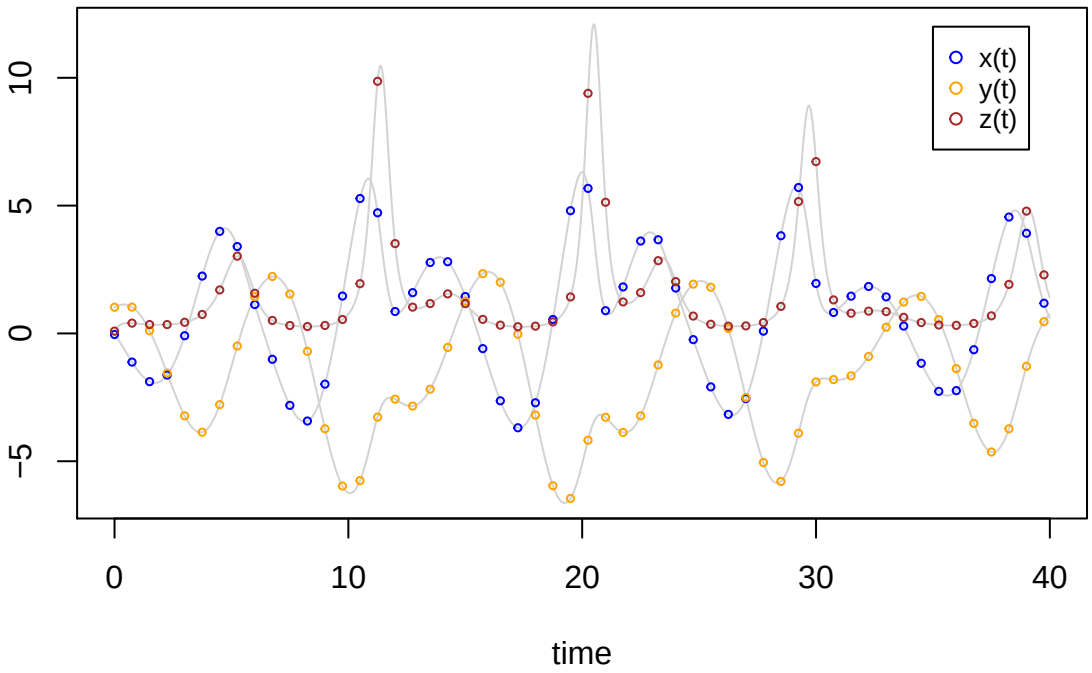

```
plot(data[,2], data[,3], type='l', xlab = 'x(t)', ylab = 'z(t)')
```



For practical reasons (noise reduction, memory size, etc.), observations may be subsampled:

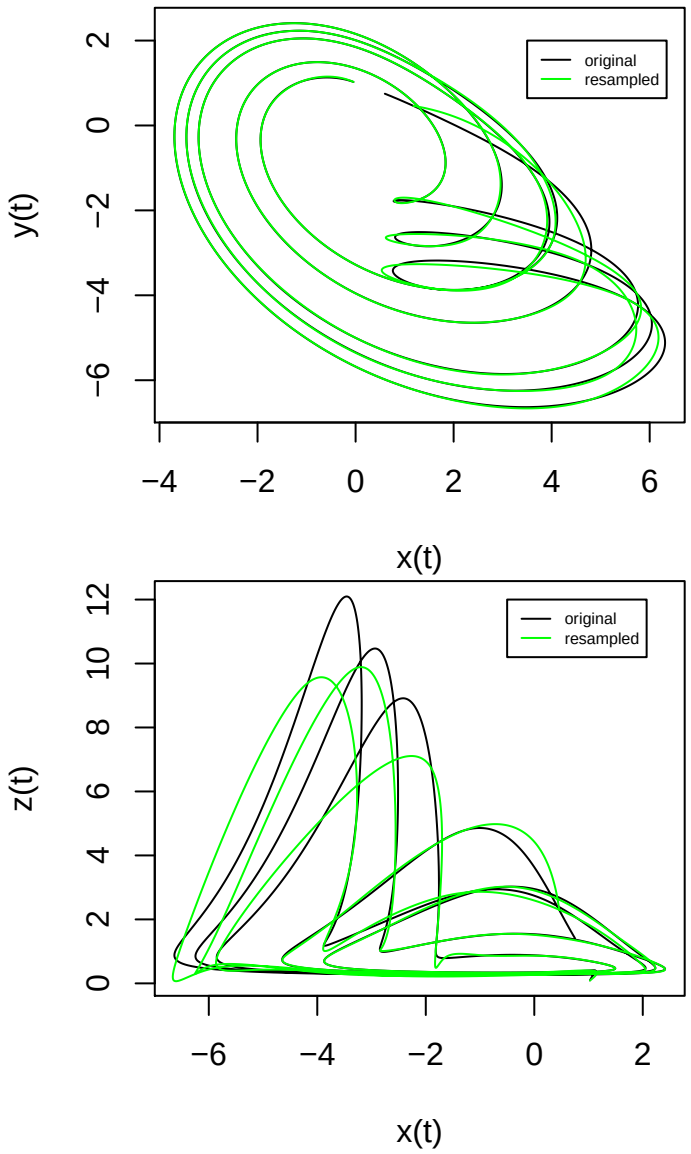
```
# plot
tin <- Ross76[,1]
data <- Ross76[,2:4]
plot(tin, data[,1], ylim = c(-6.5,12), type='l', col='lightgray', xlab = 'time', ylab = '')
lines(tin, data[,2], type='l',  col='lightgray')
lines(tin, data[,3], type='l',  col='lightgray')

# subsampled data
#
# subsampling:
Ross76us <- Ross76[seq(1,4000,by=75),]
# plot
tus <- Ross76us[,1]
datus <- Ross76us[,2:4]
lines(tus, datus[,1], type='p', cex = 0.5, col='blue')
lines(tus, datus[,2], type='p', cex = 0.5, col='orange')
lines(tus, datus[,3], type='p', cex = 0.5, col='brown')
legend(35,12,c("x(t)", "y(t)", "z(t)"), col=c('blue', 'orange', 'brown'), pch=1, cex=0.8)
```



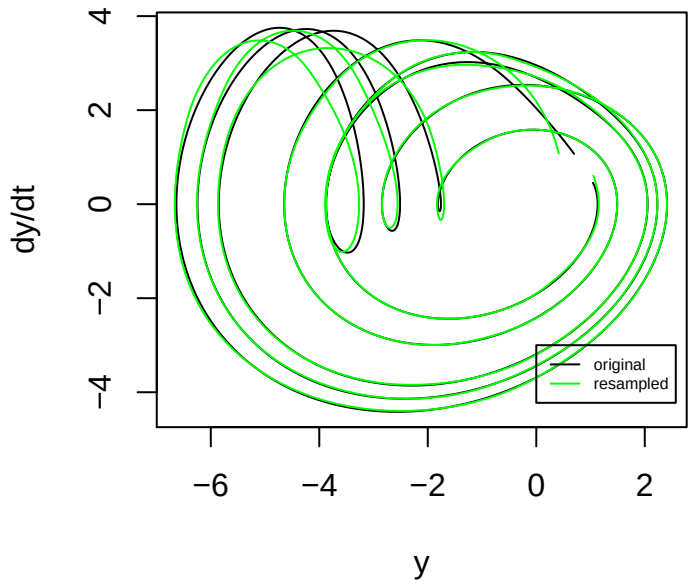
Subsampling may lead to loose an important part of the original signal. This loss may vary according to the type of observed variable (for example, several peaks of the time series are missrepresented by the subsampled signal), and this effect will become stronger when considering derivatives of higher degrees (see the first Section of the present vignette). It can be useful in such a case to resample the time series at a higher time resolution. Resampling can be performed with splines as exemplified upper for single time series. Alternative methods may be preferred depending on the complexity of the behavior under study, on the degree of subsampling, on the level of noise, etc. Here, this resampling appears efficient enough to obtain a good reconstruction of the phase space as illustrated by the following projections (x, y) and (x, z).

```
xout <- seq(min(tus), max(tus), by = 0.01)
rsplx <- spline(tus, y = datus[,1], method = "fmm", xout = xout)
rsply <- spline(tus, y = datus[,2], method = "fmm", xout = xout)
rsplz <- spline(tus, y = datus[,3], method = "fmm", xout = xout)
# plot resampled
plot(data[,1], data[,2], type='l', xlab = 'x(t)', ylab = 'y(t)')
lines(rsplx$y, rsply$y, type='l', col='green')
legend(3.5,2,c("original", "resampled"), col=c('black', 'green'), lty=1, cex=0.5)
plot(data[,2], data[,3], type='l', xlab = 'x(t)', ylab = 'z(t)')
lines(rsply$y, rsplz$y, type='l', col='green')
legend(-0.5,12,c("original", "resampled"), col=c('black', 'green'), lty=1, cex=0.5)
```



Discrepancies related to the derivatives can be easily observed considering the derivative phase portraits. The performances of resampling is obviously good for variable y:

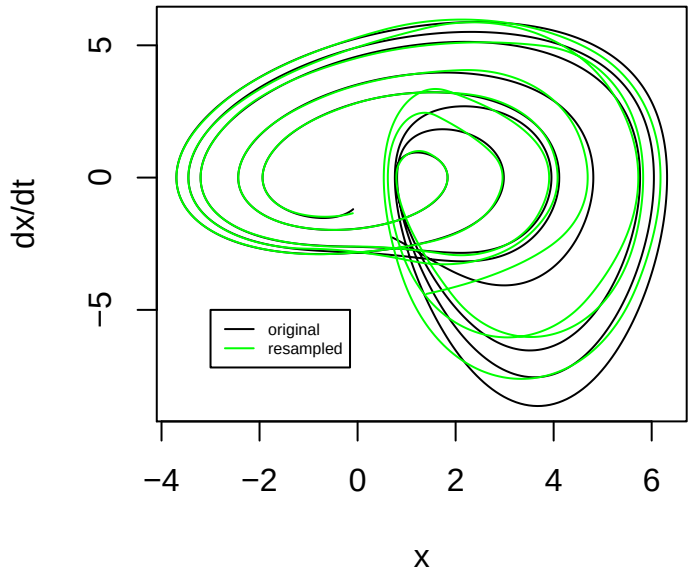
```
# derivatives
# from original signal
drv1x <- drvSucc(tin, data[,1], nDeriv=2)
drv1y <- drvSucc(tin, data[,2], nDeriv=2)
drv1z <- drvSucc(tin, data[,3], nDeriv=2)
# from resampled signal
drv3x <- drvSucc(rsplx$x, rsplx$y, nDeriv=2)
drv3y <- drvSucc(rsply$x, rsply$y, nDeriv=2)
drv3z <- drvSucc(rsplz$x, rsplz$y, nDeriv=2)
# phase portraits plot for y:
# (y, dy/dt) projection of the original signal (in black) and resampled signal (in green)
plot(drv1y$seriesDeriv[,1], drv1y$seriesDeriv[,2], type='l', xlab = 'y', ylab = 'dy/dt')
lines(drv3y$seriesDeriv[,1], drv3y$seriesDeriv[,2], type='l', col='green')
legend(0,-3,c("original", "resampled"), col=c('black', 'green'), lty=1, cex=0.5)
```

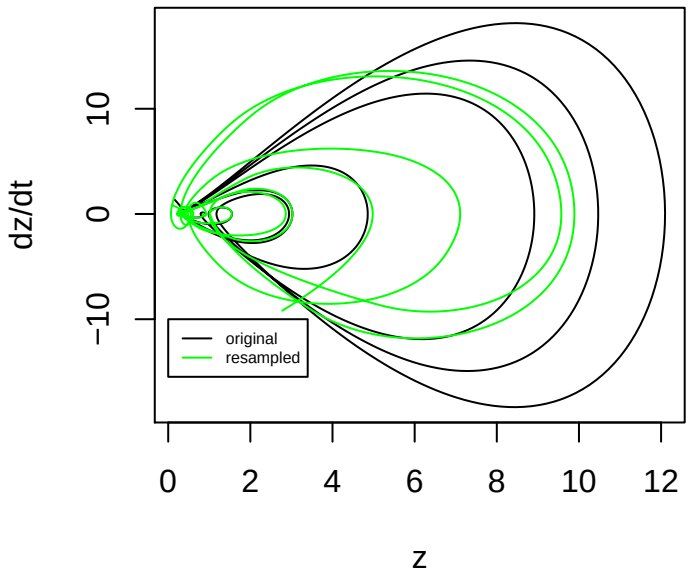


Resampling appears less efficient for variables x and z:

```
# phase portraits plot for x and z:
# (x, dx/dt) projection of the original signal (in black) and resampled signal (in green)
plot(drv1x$seriesDeriv[,1], drv1x$seriesDeriv[,2], type='l', xlab = 'x', ylab = 'dx/dt')
lines(drv3x$seriesDeriv[,1], drv3x$seriesDeriv[,2], type='l', col='green')
legend(-3,-5,c("original", "resampled"), col=c('black', 'green'), lty=1, cex=0.5)

# phase portraits plots
# (z, dz/dt) projection of the original signal (in black) and resampled signal (in green)
plot(drv1z$seriesDeriv[,1], drv1z$seriesDeriv[,2], type='l', xlab = 'z', ylab = 'dz/dt')
lines(drv3z$seriesDeriv[,1], drv3z$seriesDeriv[,2], type='l', col='green')
legend(0,-10,c("original", "resampled"), col=c('black', 'green'), lty=1, cex=0.5)
```





Conclusion and next step

Although resampling with splines may not permit to reconstruct the original signal perfectly, it is found that resampling can enable to obtain models producing a dynamical behavior very similar to the dynamics of the original dynamical system. In practice, it can be an efficient way to deal with subsampling. This type of resampling could be efficiently used for applying the global modelling technique both from single[6], and multiple time series[7, 8].

Once the time series have been carefully prepared, the global modelling technique can be applied. This next step will be presented in the following vignette `3 Modelling`.

[6]S. Mangiarotti, L. Drapeau, & C. Letellier, 2014. Two chaotic global models for cereal crops cycles observed from satellite in Northern Morocco, *Chaos*, **24**, 023130.

[7]S. Mangiarotti, 2015. Low dimensional chaotic models for the plague epidemic in Bombay (1896-1911), *Chaos, Solitons & Fractals*, **81**(A), 184-196.

[8]S. Mangiarotti, M. Peyre & M. Huc, 2016. A chaotic model for the epidemic of Ebola virus disease in West Africa (2013-2016). *Chaos*, **26**, 113112.