A quick guide to multisensi, an R package for multivariate sensitivity analyses

Caroline Bidot, Hervé Monod, Marie-Luce Taupin

MaIAGE, INRA, Université Paris-Saclay, 78350 Jouy-en-Josas, France

July 12, 2017

Contents

1 Introduction

Global sensitivity analysis is an essential tool for modellers in all application areas. Its aim is to quantify and compare the influence of uncertain parameters (or other input variables) on the output of a given model. There exist many different methods to perform sensitivity analysis, but they are usually restricted to a single output variable. On the contrary, the R package multisensi is specifically designed to perform global sensitivity analyses on a multivariate model output. It calculates and represents graphically sensitivity indices on each output variable or on combinations of output variables arising from dimension reduction techniques.

The initial version of multisensi was based on the combination of *(i)* a factorial design on the uncertain model parameters; *(ii)* the application of principal components analysis on the model output; *(iii)* anova-based sensitivity analyses on the first principal components. This idea was proposed by Campbell *et al.* [1] and further studied by Lamboni *et al.* [5] (see also [11]). It is still implemented in multisensi, but the present version includes alternative methods to perform the dimension reduction : splines, bsplines and polynomial regression. In addition, several methods of global sensitivity analysis can now be used, including those implemented in the package sensitivity [7].

Applications in agronomy and epidemiology are presented by Lamboni *et al.* [4] and Lurette *et al.* [6]. For a detailed introduction to the methods of global sensitivity analysis, we refer to Saltelli *et al.* [9] and Faivre *et al.* [2] (in French). Multivariate techniques are presented, for example, in James *et al.* [3].

2 Case study: the Verhulst model of population dynamics

In the following, we illustrate the methods implemented in multisensi using a very simple model, the dynamic population model of Verhulst.

Case study specifications The Verhulst model is given by the equation

$$Y_t = \frac{K}{1 + (K/Y_0 - 1)\exp(-at)},$$

where Y_t is the population size at time t and Y_0, K, a are respectively the initial size, the carrying capacity and the rate of maximum population growth. The aim in our case study is to evaluate the influence of these three parameters on the population sizes until time $T = 100$. For this, simulated population sizes are recorded at times $5, 10, \cdots, 100$. The parameter uncertainty ranges of interest are assumed to be $(100, 1000)$ for K, $(1, 40)$ for Y_0, $(0.05, 0.2)$ for a.

Model implementation The R function `verhulst` is created to run the model for given values of K, a, Y_0 and for a vector t of output times. The output of `verhulst` is the vector of population sizes at the times in t.

```r
verhulst <- function(K, Y0, a, t) {
    output <- K/(1 + (K/Y0 - 1) * exp(-a * t))
    return(output)
}
```

Since the methods implemented in multisensi require to run the dynamic population model repeatedly, another function called `verhulst2` is created. It takes a dataframe of input combinations as its first argument and the time steps of interest as its second argument..

```r
T <- seq(from = 5, to = 100, by = 5)
verhulst2 <- function(X, t = T) {
    out <- matrix(nrow = nrow(X), ncol = length(t), NA)
    for (i in 1:nrow(X)) {
        out[i, ] <- verhulst(X$K[i], X$Y0[i], X$a[i], t)
    }
    out <- as.data.frame(out)
    names(out) <- paste("t", t, sep = "")
    return(out)
}
```

A sample of population dynamics The output of the Verhulst model is plotted in Fig. 1, for a few combinations of values of the parameters.

```
n <- 10
set.seed(1234)
X <- data.frame(K = runif(n, min = 100, max = 1000), Y0 = runif(n, min = 1,
    max = 40), a = runif(n, min = 0.05, max = 0.2))
Y <- verhulst2(X)
par(cex.axis = 0.7, cex.lab = 0.8)
plot(T, Y[1, ], type = "l", xlab = "Time", ylab = "Population size",
    ylim = c(0, 1000))
for (i in 2:n) {
    lines(T, Y[i, ], type = "l", col = i)
}
```

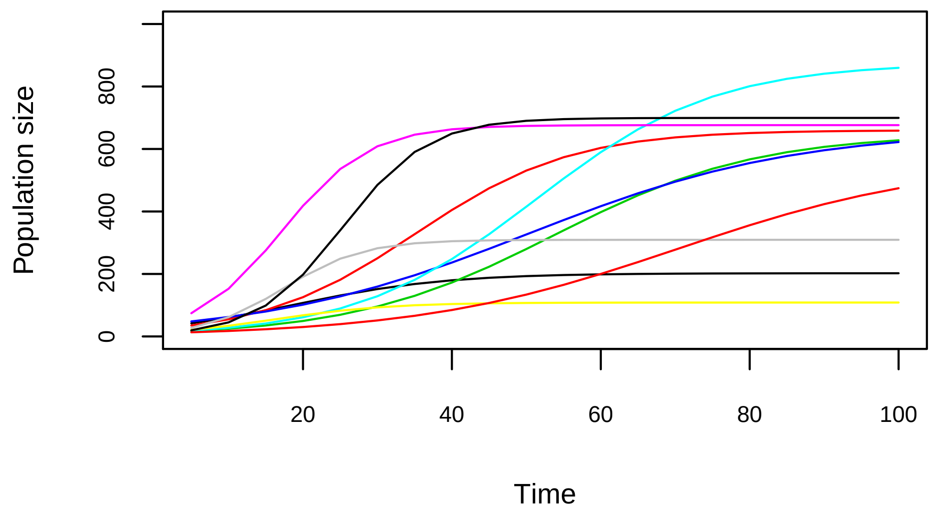


Figure 1: Population size versus time according to the Verhulst model for 10 combinations of values of K, Y_0 and a.

3 Sequential univariate sensitivity analyses

3.1 Calculation of sensitivity indices

Now we want to perform sensitivity analyses on the population sizes with respect to the three uncertain parameters: K, Y_0, and a. This can be done in different ways by using the main function of the package, which is called `multisensi` like the package itself.

A first and obvious option is to perform separate sensitivity analyses at $t = 5, 10, ..., 100$. To do this, `multisensi` must be used with the argument `reduction=NULL` :

```
library(multisensi)
verhulst.seq <- multisensi(model=verhulst2, reduction=NULL, center=FALSE,
  design.args = list( K=c(100,400,1000), Y0=c(1,20,40), a=c(0.05,0.1,0.2)))

## [*] Design
## [*] Response simulation
## [*] Analysis + Sensitivity Indices
```

By keeping the default values of `multisensi` for the design and analysis arguments, a full factorial design is created according to the factors' levels provided in the design.args argument. The results are then analysed by the R function aov, with an anova formula including main effects and two-factor interactions.

The result verhulst.seq is an object of class dynsi, which has specific print, summary, and plot methods. The print and summary functions give the sensitivity indices :

```
print(verhulst.seq, digits = 2)

##
##  Main sensitivity indices
##       t5   t10   t15  t20  t25  t30  t35   t40   t45   t50   t55
## K  0.014 0.053 0.096 0.13 0.18 0.23 0.30 0.351 0.393 0.433 0.476
## a  0.092 0.210 0.257 0.27 0.28 0.29 0.30 0.287 0.258 0.230 0.206
## Y0 0.818 0.519 0.335 0.24 0.20 0.16 0.12 0.087 0.073 0.068 0.064
##     t60   t65   t70   t75   t80   t85   t90   t95  t100   GSI
## K  0.52 0.573 0.622 0.664 0.698 0.725 0.748 0.768 0.787 0.562
## a  0.18 0.165 0.146 0.128 0.111 0.096 0.082 0.069 0.058 0.165
## Y0 0.06 0.053 0.046 0.039 0.034 0.030 0.027 0.025 0.023 0.063
##
##  Total sensitivity indices
##       t5  t10  t15  t20  t25  t30  t35  t40  t45  t50  t55  t60
## K  0.035 0.14 0.25 0.34 0.41 0.48 0.55 0.59 0.61 0.64 0.67 0.70
## a  0.148 0.35 0.45 0.49 0.49 0.50 0.51 0.51 0.48 0.44 0.40 0.36
## Y0 0.878 0.66 0.50 0.40 0.33 0.25 0.18 0.15 0.14 0.14 0.13 0.13
##     t65  t70   t75   t80   t85   t90   t95  t100  GSI
## K  0.73 0.77 0.790 0.808 0.822 0.832 0.842 0.853 0.71
## a  0.32 0.29 0.259 0.232 0.209 0.187 0.167 0.148 0.32
## Y0 0.11 0.10 0.091 0.086 0.083 0.082 0.081 0.079 0.13
```

3.2 Graphical representation

Rather than reading tables of sensitivity indices, the user may prefer to plot them. Fig. 2 shows the graphics obtained by the following code, where the two `plot` commands differ only by the `normalized` argument :

```
# color palettes: rainbow, heat.colors, terrain.colors, topo.colors,
# cm.colors
plot(verhulst.seq, normalized = TRUE, color = terrain.colors, gsi.plot = FALSE)
title(xlab = "Time in half-decades.")
plot(verhulst.seq, normalized = FALSE, color = terrain.colors, gsi.plot = FALSE)
title(xlab = "Time in half-decades.")
```

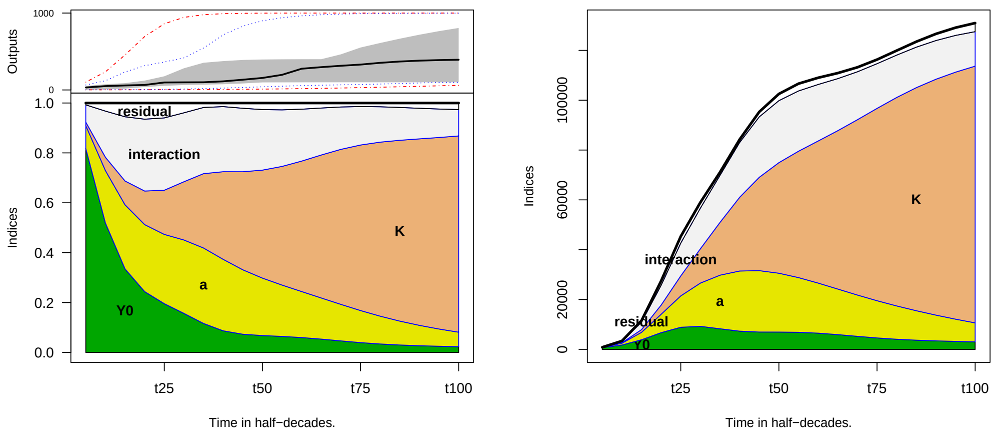


Figure 2: Dynamics of the sensitivity indices of the Verhulst model from $t = 5$ to 100, with indices normalized either to 1 (left panel) or to the variance at each time t (right panel). In the left panel, the upper subplot shows the extreme (tirets), inter-quartile (grey) and median (bold line) output values at all time steps. The lower subplot represents the sensitivity indices at all time steps for the main effects and the first-order interactions.

In the lower subplot of Fig. 2 (left panel), the sensitivity indices at time t are given by the lengths of the different colors along the vertical bar at time t. Thus one can see clearly that the population size at time $t = 25$ is sensitive to the main effects of a, Y_0, K in quite similar proportions, with interactions accounting for roughly one-fourth of the variability. At time $t = 100$, the population size is sensitive mainly to K, which is logical since the population sizes are close to their carrying capacity K. The upper subplot illustrates how output quantiles vary along the time steps. This is useful to avoid over-interpretation of the sensitivity indices at times when the variability between simulations is low. For example, the population size is very sensitive to Y_0 in the first time steps, but then the variability between simulations is still very low as shown in the upper subplot. This can also be seen by setting the argument `normalized=FALSE` as in Fig. 2 (right panel). In that case, the sensitivity indices at time t are constrained to sum to the output variance at time t rather than to reflect proportions summing to one at all times t.

3.3 Calculating simulations apart

If needed, the design or the model output can be calculated apart from the `multisensi` function. This allows to use a design generated outside `multisensi` or to apply `multisensi` to simulated data based on a code implemented outside R. Thus, the commands below are equivalent to those in Section 3.1 :

```
X <- expand.grid(K=c(100,400,1000), Y0=c(1,20,40), a=c(0.05,0.1,0.2))
Y <- verhulst2(X) ## this part can be performed outside R if necessary
verhulst.seq <- multisensi(design=X, model=Y, reduction=NULL, center=FALSE)

## [*] Analysis + Sensitivity Indices
```

4 Multivariate sensitivity analysis based on PCA

The sequential sensitivity analyses of Section 3 give interesting information on the evolution of parameters' influence over time. However other methods based on multivariate techniques can give a more synthetic view of the parameters' impact on the population dynamics. Their common principle is to proceed to the analysis in two steps:

1. **dimension reduction:** a multivariate technique is applied to decompose the simulated dynamics on a reduced basis of d canonical dynamics, which will be called the *basic trajectories*. The multivariate techniques available in multisensi are principal components analysis (PCA), B-splines, orthogonal B-splines or projection on a polynomial basis;

2. **sensitivity analysis:** for each basic trajectory, sensitivity analysis is applied to the associated coefficients of the decomposition.

Mathematically, the population dynamics are decomposed into

$$Y_{i,t} = \sum_{j=1}^{d} \alpha_{ij} Z_{j,t} + \eta_{i,t}, \tag{1}$$

where $Y_{i,t}$ is the output of simulation i at time t, $\left(Z_{j,t}\right)_{t=1,\cdots,T}$ is the jth basic trajectory (which depends on the multivariate technique), α_{ij} is the coefficient of simulation i associated with the jth basic trajectory, d is the reduction dimension and $\eta_{i,t}$ is the approximation error on simulation i at time t due to dimension reduction. Sensitivity analyses are applied to the α_{ij} coefficients, for each basic dimension $j = 1, \cdots, d$. In addition, it is possible to calculate the generalised sensitivity indices (GSI), which are weighted means of the sensitivity indices over the d dimensions, and the global criterion (GC), which quantifies the proportion of variability accounted for by the approximation resulting from both the dimension reduction and the restriction to low-order factorial effects in the sensitivity analysis (see Lamboni *et al.* [5] and [11]).

In this Section, we illustrate this approach by focusing on principal components analysis (PCA). For any given reduced dimension d, the PCA decomposition maximises the variability between dynamics taken into account by the summation in equation (1). Let V denote the variance-covariance matrix of the T vectors of simulated output variables $\left(Y_{i,t}\right)_{i=1,\cdots,N}$, where N is the size of the design; then the PCA basic trajectories $\left(Z_{j,t}\right)_{t=1,\cdots,T}$ are the eigenvectors of V in decreasing order of their associated eigenvalues.

4.1 Calculation

Multivariate sensitivity analyses can be performed by using the same multisensi function as before. The choice of the multivariate technique is now specified by the argument `reduction=basis.ACP` :

```
## Note that this old syntax of multisensi still works:
## verhulst.gsi <- gsi(formula=2, Y, X)
verhulst.pca <- multisensi(design=X, model=Y, reduction=basis.ACP, scale=FALSE)

## [*] Dimension Reduction
## [*] Analysis + Sensitivity Indices
## [*] Goodness of fit computing
```

By keeping the default argument `dimension = 0.95`, the dimension d is selected automatically as the smallest value such that 95% of the total variability (or inertia) between dynamics is taken into account. Another option would be to specify `dimension = 3`, for example. Two other arguments are associated with dimension reduction, center and scale. When the output is a time series with the same variable calculated from time $t = 1$ to $t = T$, as in our Verhulst model case study, it can be more pertinent to keep the raw ranges of variation of Y over time. This is why we use the argument `scale=FALSE`. Note that we choose to use again the factorial design X and the simulated output Y calculated in Section 3.3. By default, the sensitivity analyses are performed by anova with a formula including main effects and two-factor interactions, as in Section 3.

The result verhulst.pca is an object of class gsi, which has print, summary and plot methods associated, in addition to the more specific functions graph.pc and graph.bar. The summary gives information on the inertia proportions explained by the first principal components, together with the tables of sensitivity indices.

```
summary(verhulst.pca, digits = 2)

##
##  Components cumulated inertia percentages and Global Criterion
##  PC1 PC2  GC
##  90  98  96
##
##  Main sensitivity indices
##      PC1  PC2   GSI
## K  0.610 0.15 0.573
## a  0.159 0.27 0.168
## Y0 0.059 0.10 0.063
##
##  Total sensitivity indices
##     PC1  PC2  GSI
## K  0.76 0.33 0.73
## a  0.29 0.59 0.32
## Y0 0.11 0.30 0.12
##
##  gsi outputs
##               Design             Response Principal Components
##                  "X"                  "Y"                  "H"
##             Loadings        PCs variances              Indices
##                  "L"             "lambda"                 "SI"
##  First order indices        Total indices  Interaction indices
##                "mSI"                "tSI"                "iSI"
##          Correlation              Inertia              Rsquare
##                "cor"            "inertia"            "Rsquare"
##         Informations
##          "call.info"
```

Here the inertia proportions show that the first two principal components explain 98% of the total variability between the calculated population dynamics. So by default, the multisensi function performs the sensitivity analyses for these first two components only. The first component is influenced mainly by K (SI=0.61), while the second component is influenced mainly by a, although less strongly. The influence of Y_0 is exerted mainly through interactions with K and a. The last column in the tables of sensitivity indices gives the generalised sensitivity indices (GSI), which are the averages of the PC1 and PC2 indices weighted by the PC percentages of inertia.

4.2 Graphical representation

When a reduction technique is applied, several types of graphical representation can be useful to interpret the results. This is why the `plot` method allows for several options to be used depending on the value given to the `graph` argument.

The `graph`=1 option gives multipanel graphics as shown in Fig. 3. There is one column per dimension j and two rows. The aim of the upper row is to show what the components in dimension j look like with respect to the output variables, so they present quantile curves $\left(\alpha_{Qj} Z_{j,t}\right)_{t=1,\cdots,T}$, where α_{Qj} denotes the quantile Q of the α_{ij} values for $i = 1, \cdots, N$ (see details in the legend of Fig. 3). The lower row contains the bar plot of sensitivity indices for each dimension j of interest.

```r
plot(verhulst.pca, graph = 1)
```

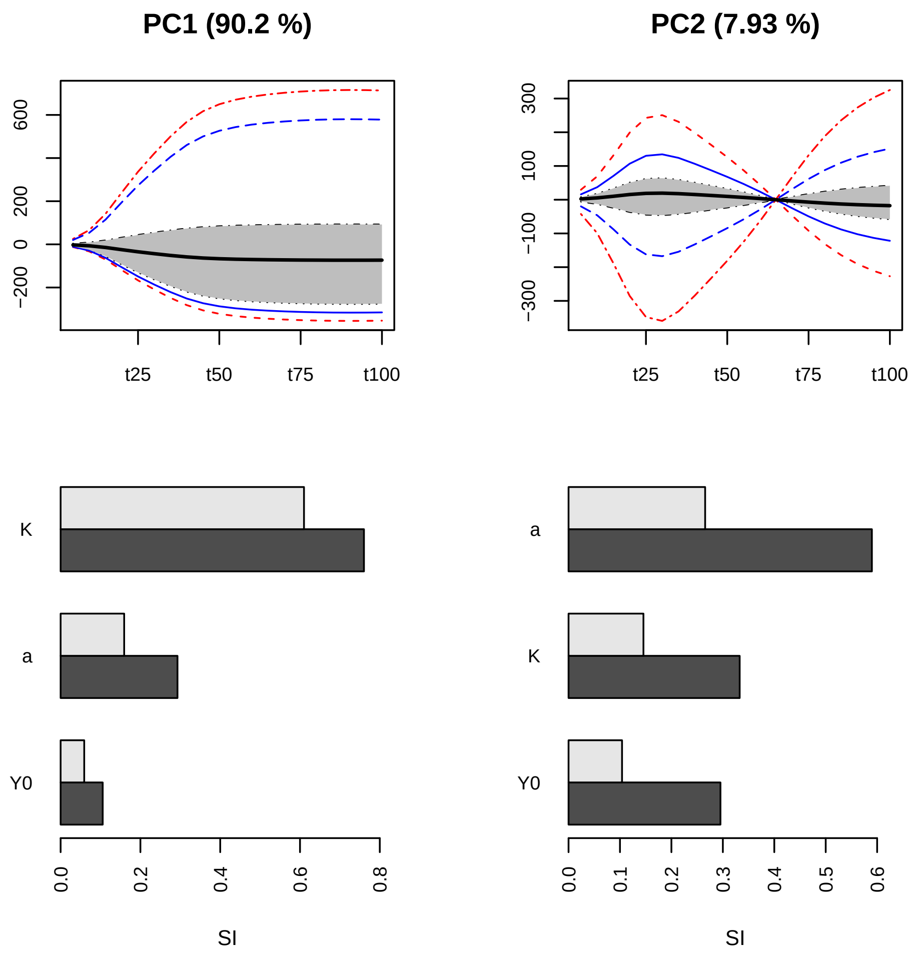


Figure 3: Plots for the PCA multivariate sensitivity analysis of the Verhulst model. Upper subplots: functional boxplots of the principal components, with time on the x-axis and population size contribution on the y-axis (red curves: extreme values, blue: 1/10 and 9/10 percentiles, grey area: inter-quartile, black: median). Lower subplots: sensitivity indices (light grey: first order indices, dark grey: total indices).

For the Verhulst case study, the left upper plot in Fig. 3 shows that the first principal component, which explains more than 90% of the inertia, captures essentially an average effect over time. The left lower plot confirms that this effect is mainly influenced by the maximum population size K, with smaller contributions of a, Y_0 and interactions.

The right upper plot in Fig. 3 shows that the second principal component, which explains about 8% of the inertia, captures the contrast in the dynamics between the early and late periods. The right lower plot shows that this effect is influenced first by a, but also by K and Y_0 and by strong two-factor interactions.

The graph=2 option gives the bar plot of generalised sensitivity indices (see Fig. 4). In the case study, these indices represent the contributions of the factors to the whole variability between the dynamics of population size.

```
plot(verhulst.pca, graph = 2)
```

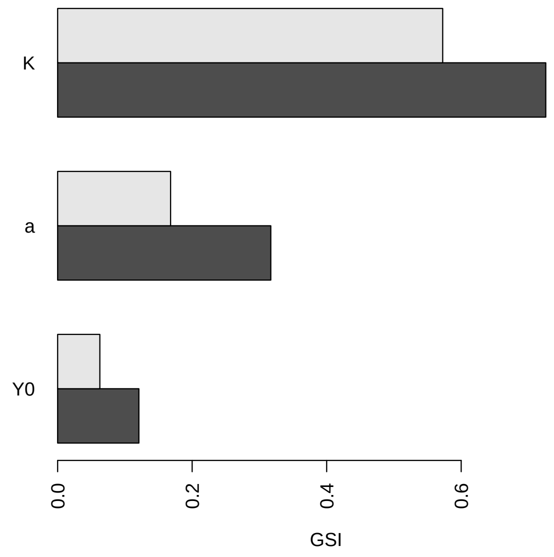


Figure 4: Bar plot of the PCA generalised sensitivity indices of the Verhulst model.

For each output variable, the multivariate sensitivity analysis accounts only for part of the variability between simulations. The first reason is due to dimension reduction. The second reason is because the sensitivity analysis is usually restricted to the factorial effects of first order or possibly first and second order. In multisensi, these proportions of variability that are accounted for can be quantified by R^2 coefficients of determination, provided the sensitivity analysis is based on anova. The graph=3 option allows to plot these R^2 coefficients of determination, as shown in Fig. 5 for the case study. The R^2 values are low for the first output variables, because they have low variance and so have little weight in the determination of the basic trajectories. The R^2 values would have been more uniform if we had chosen to normalise the output variables by setting scale=TRUE.

```r
plot(verhulst.pca, graph = 3)
```

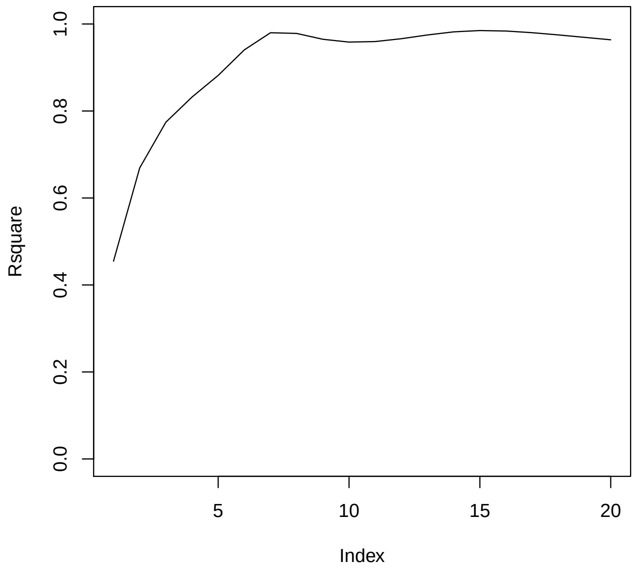


Figure 5: Coefficients of determination of the output variables for the PCA multivariate sensitivity analysis of the Verhulst model on raw output data.

5 Alternative reduction techniques

In practice, it can be useful to apply sensitivity analysis with multivariate techniques other than PCA. The package multisensi provides projection methods either on a reduced polynomial basis or on data-based spline bases.

5.1 Polynomial reduction of the multivariate output

In the case of a polynomial decomposition, the $(Y_{i,t})_{t=1,\cdots,T}$ dynamics are approximated by polynomials of a given degree $d-1$. Thus the $(Z_{j,t})_{t=1,\cdots,T}$ basic trajectories of equation (1) are the monomials of t of degree up to $d-1$.

In multisensi, this decomposition is obtained by setting the argument reduction = basis.poly. It is necessary to give additional information through the basis.args argument. We do it here for the Verhulst model by specifying that the polynomial must be of degree six or lower and by giving the vector of output time coordinates $t = (5, \cdots, 100)^T$.

```r
verhulst.poly <- multisensi(design = X, model = Y, reduction = basis.poly,
      dimension = 0.99, center = FALSE, scale = FALSE, cumul = FALSE,
      basis.args = list(degree=6, x.coord=T), analysis = analysis.anoasg,
      analysis.args = list(formula=2, keep.outputs=FALSE))

## [*] Dimension Reduction
## [*] Analysis + Sensitivity Indices
## [*] Goodness of fit computing

summary(verhulst.poly, digits=2)
```

```
##
##  Components cumulated inertia percentages and Global Criterion
##  Deg0 Deg1 Deg2 Deg3   GC
##   76   93   98   99   97
##
##  Main sensitivity indices
##     Deg0   Deg1  Deg2   Deg3   GSI
## K  0.571 0.7327 0.093 0.0047 0.567
## a  0.179 0.0526 0.331 0.2838 0.166
## Y0 0.078 0.0014 0.049 0.0902 0.064
##
##  Total sensitivity indices
##    Deg0 Deg1 Deg2 Deg3  GSI
## K  0.73 0.79 0.39 0.22 0.72
## a  0.31 0.20 0.71 0.64 0.32
## Y0 0.12 0.10 0.21 0.32 0.13
##
##  gsi outputs
##               Design             Response Principal Components
##                  "X"                  "Y"                  "H"
##             Loadings        PCs variances              Indices
##                  "L"             "lambda"                 "SI"
##  First order indices        Total indices  Interaction indices
##                "mSI"                "tSI"                "iSI"
##          Correlation              Inertia              Rsquare
##                "cor"            "inertia"            "Rsquare"
##         Informations
##          "call.info"
```

The results show that polynomials of degree 3 are sufficient to integrate more than 99% of the output variability. Graphics are given in Fig. 6.

```
plot(verhulst.poly, nb.comp = 3, graph = 1)
```

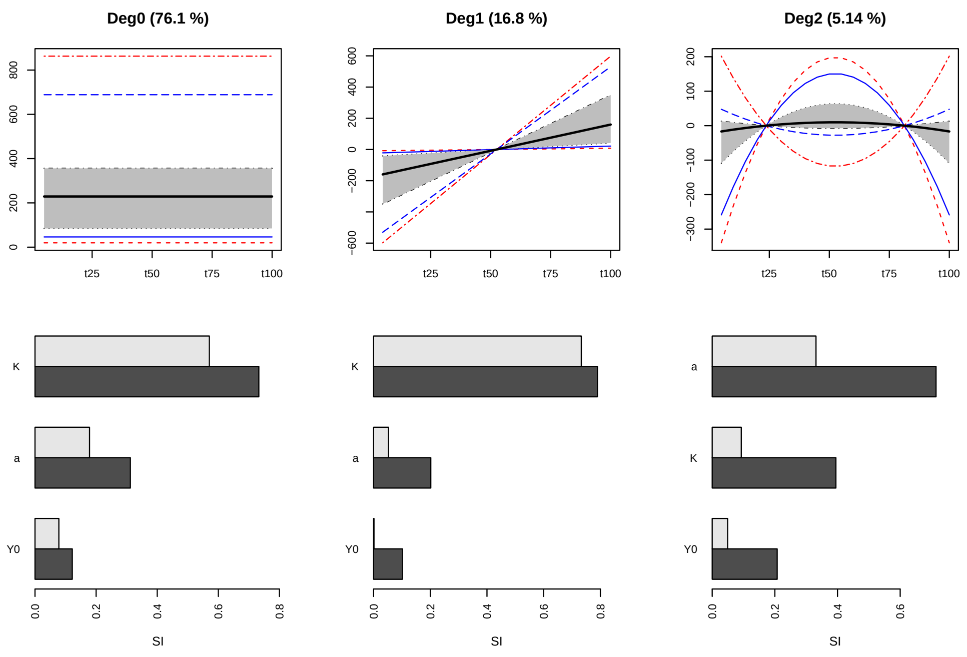


Figure 6: Plots for the multivariate sensitivity analysis based on polynomial regression. Upper subplots: functional boxplots of the constant, linear and quadratic components, with time on the x-axis and population size contribution on the y-axis (red curves: extreme values, blue: 1/10 and 9/10 percentiles, grey area: inter-quartile, black: median). Lower subplots: sensitivity indices (light grey: first order indices, dark grey: total indices).

5.2 Spline modelling of the population dynamics

Splines represent other popular techniques of multivariate analysis (see [3, 10, 8]). We give an example just below with B-splines, see also Fig. 7. More information is given in the help of basis.bsplines.

```
## bsplines
verhulst.bspl <- multisensi(design=X, model=Y, reduction=basis.bsplines,
                        dimension=NULL, center=FALSE, scale=FALSE,
                        basis.args=list(knots=10, mdegree=3), cumul=FALSE,
                        analysis=analysis.anoasg,
                        analysis.args=list(formula=2, keep.outputs=FALSE))

## [*] Dimension Reduction
## [*] Analysis + Sensitivity Indices
## [*] Goodness of fit computing
```

```
plot(verhulst.bspl, nb.comp = 5, graph = 1)
```

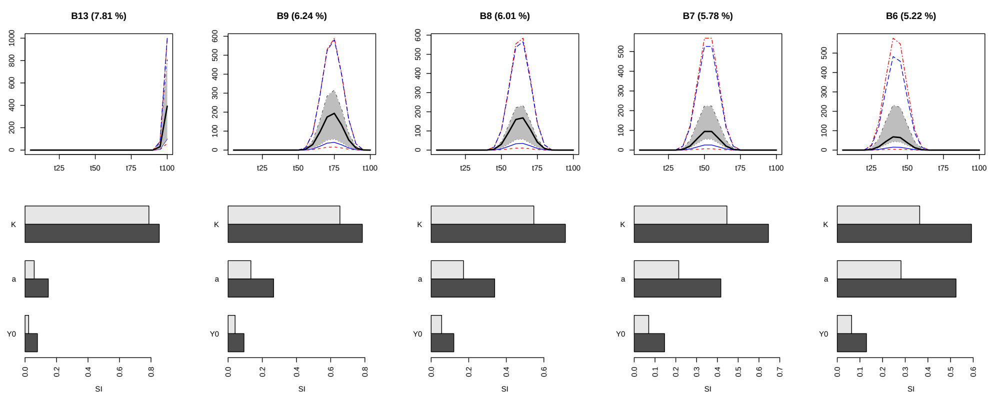


Figure 7: Plots for the multivariate sensitivity analysis based on bspline regression. Upper subplots: functional boxplots of the b-splines, with time on the x-axis and population size contribution on the y-axis (red curves: extreme values, blue: 1/10 and 9/10 percentiles, grey area: inter-quartile, black: median). Lower subplots: sensitivity indices (light grey: first order indices, dark grey: total indices).

6 Alternative methods of sensitivity analysis

Sensitivity analyses based on factorial design plus anova are very useful and convenient. However, they oblige to restrict the levels of each input factor to a few discretised values from their uncertainty ranges. Other methods of global sensitivity analysis take better account of continuous ranges of input values. Many of them can be used with multisensi.

6.1 With Sobol2007 implemented in the package **sensitivity**

The function multisensi is compatible with the methods of sensitivity analysis implemented in the sensitivity package. To illustrate this, we first call the sensitivity package:

```
library(sensitivity)
```

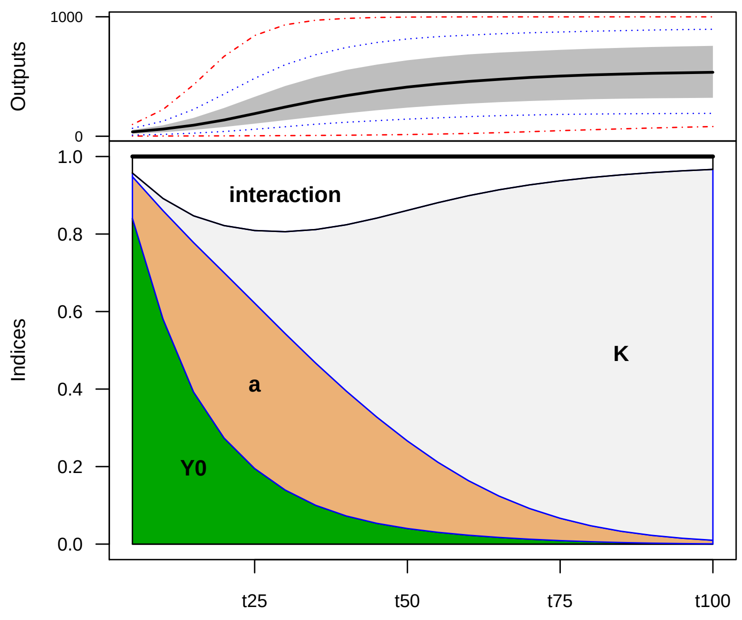
Then we use the method sobol2007 of the sensitivity package :

```
m <- 10000
Xb <- data.frame(K = runif(m, min = 100, max = 1000), Y0 = runif(m, min = 1,
    max = 40), a = runif(m, min = 0.05, max = 0.2))
verhulst.seq.sobol <- multisensi(design = sobol2007, model = verhulst2,
    reduction = NULL, analysis = analysis.sensitivity, center = TRUE,
    design.args = list(X1 = Xb[1:(m/2), ], X2 = Xb[(1 + m/2):m, ], nboot = 100),
    analysis.args = list(keep.outputs = FALSE))

## [*] Design
## [*] Response simulation
## [*] Analysis + Sensitivity Indices

print(verhulst.seq.sobol, digits = 2)

##
##  Main sensitivity indices
##        t5   t10   t15  t20  t25  t30  t35   t40   t45  t50  t55
## K  0.0093 0.031 0.069 0.12 0.19 0.26 0.34 0.429 0.513 0.59 0.67
## a  0.1075 0.280 0.386 0.43 0.43 0.40 0.37 0.322 0.274 0.23 0.18
## Y0 0.8405 0.581 0.393 0.27 0.20 0.14 0.10 0.073 0.054 0.04 0.03
##      t60   t65   t70   t75    t80   t85    t90    t95   t100   GSI
## K  0.735 0.790 0.835 0.871 0.8984 0.920 0.9359 0.9480 0.9568 0.704
## a  0.141 0.107 0.079 0.058 0.0412 0.029 0.0201 0.0138 0.0094 0.153
## Y0 0.023 0.017 0.013 0.009 0.0062 0.004 0.0024 0.0013 0.0005 0.042
##
##  Total sensitivity indices
##       t5   t10  t15  t20  t25  t30  t35  t40   t45   t50   t55
## K  0.013 0.061 0.14 0.23 0.33 0.42 0.50 0.58 0.650 0.714 0.771
## a  0.152 0.380 0.52 0.58 0.58 0.55 0.51 0.45 0.391 0.329 0.269
## Y0 0.877 0.671 0.50 0.36 0.26 0.19 0.14 0.10 0.082 0.068 0.056
##      t60  t65   t70   t75   t80   t85   t90   t95  t100   GSI
## K  0.818 0.86 0.890 0.916 0.936 0.952 0.963 0.972 0.979 0.783
## a  0.214 0.17 0.128 0.096 0.071 0.051 0.037 0.026 0.017 0.222
## Y0 0.047 0.04 0.034 0.028 0.024 0.020 0.016 0.013 0.010 0.068
```

Results are given above and the sequence of sensitivity indices is displayed in Fig. 8. The results are very similar to those of Section 3, but here they have obtained by varying the input factors across their whole uncertainty intervals.

Of course, the Sobol method of sensitivity analysis can also be combined with the multivariate techniques (PCA, polynomials, splines), by changing the reduction argument.

```r
plot(verhulst.seq.sobol, normalized = TRUE, color = terrain.colors)
```



```r
title(xlab = "Time in half-decades")
```

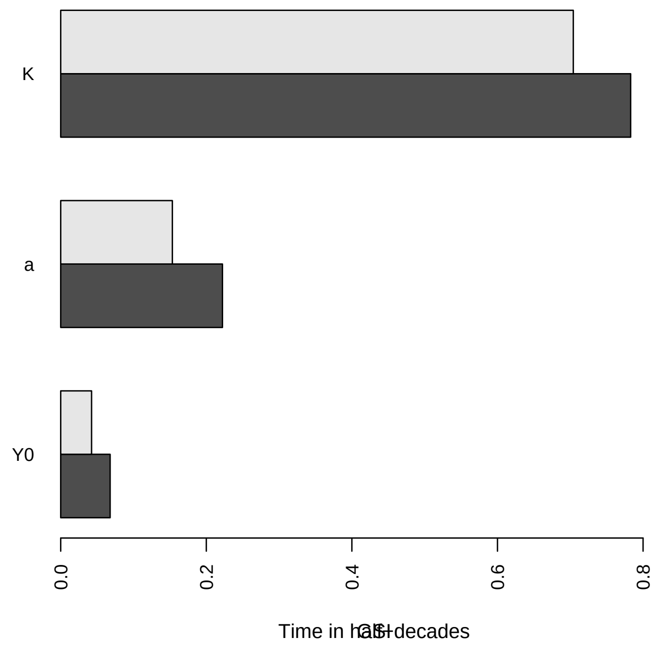


Figure 8: Evolution of the Sobol sensitivity indices of the Verhulst model from $t = 5$ to $t = 100$. The upper subplot shows the extreme (tirets), inter-quartile (grey) and median (bold line) output values at all time steps. The lower subplot represents the sensitivity indices at all time steps for the main effects and the first-order interactions.

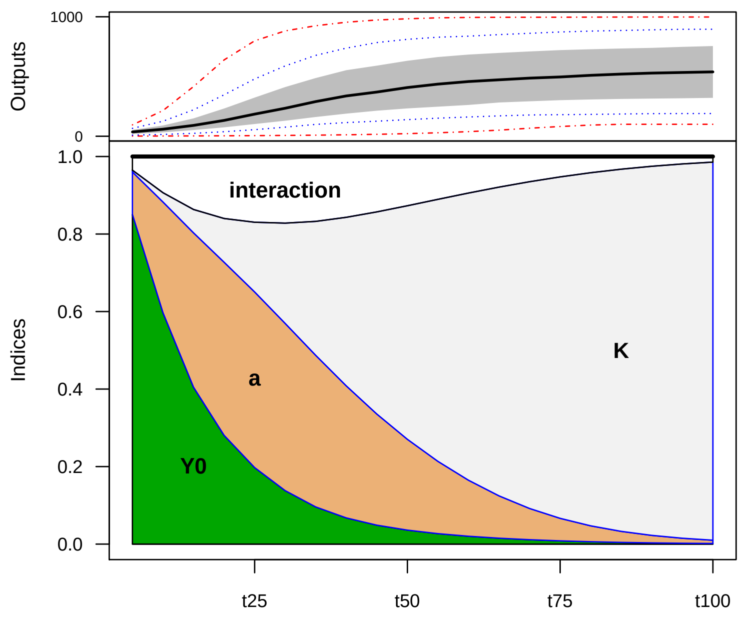
6.2 With fast99 implemented in the package **sensitivity**

Another possibility is to use the method fast99 :

```
verhulst.seq.fast <- multisensi(design = fast99, model = verhulst2,
      center = FALSE, reduction = NULL, analysis = analysis.sensitivity,
      design.args=list( factors=c("K","Y0","a"), n=1000, q = "qunif",
        q.arg = list(list(min=100, max=1000), list(min=1, max=40),
          list(min = 0.05, max = 0.2))),
      analysis.args=list(keep.outputs=FALSE))

## [*] Design
## [*] Response simulation
## [*] Analysis + Sensitivity Indices

  print(verhulst.seq.fast,digits=2)

##
##  Main sensitivity indices
##        t5   t10  t15  t20  t25  t30   t35   t40   t45   t50   t55
## K  0.0047 0.024 0.06 0.11 0.18 0.26 0.346 0.435 0.522 0.603 0.676
## a  0.1094 0.286 0.40 0.45 0.45 0.43 0.391 0.341 0.287 0.234 0.186
## Y0 0.8507 0.597 0.40 0.28 0.20 0.14 0.096 0.068 0.049 0.036 0.027
##     t60   t65   t70    t75    t80    t85    t90    t95   t100   GSI
## K  0.74 0.797 0.843 0.8809 0.9110 0.9345 0.9523 0.9657 0.9755 0.714
## a  0.14 0.109 0.081 0.0581 0.0410 0.0284 0.0194 0.0130 0.0085 0.159
## Y0 0.02 0.015 0.011 0.0083 0.0061 0.0044 0.0031 0.0022 0.0015 0.041
##
##  Total sensitivity indices
##       t5   t10  t15  t20  t25  t30  t35  t40   t45   t50   t55
## K  0.011 0.056 0.14 0.23 0.32 0.41 0.50 0.57 0.645 0.710 0.767
## a  0.141 0.364 0.51 0.57 0.58 0.57 0.53 0.48 0.417 0.354 0.292
## Y0 0.884 0.675 0.50 0.36 0.26 0.19 0.14 0.11 0.091 0.078 0.069
##      t60   t65   t70   t75   t80   t85   t90   t95  t100   GSI
## K  0.817 0.859 0.893 0.921 0.942 0.959 0.971 0.980 0.987 0.785
## a  0.236 0.187 0.146 0.113 0.086 0.065 0.049 0.036 0.026 0.239
## Y0 0.061 0.054 0.047 0.041 0.035 0.029 0.024 0.019 0.015 0.076
```

Results are shown above and the sequence of sensitivity indices is displayed in Fig. 9.

```
plot(verhulst.seq.fast, normalized = TRUE, color = terrain.colors)
```



```
title(xlab = "Time in half-decades")
```

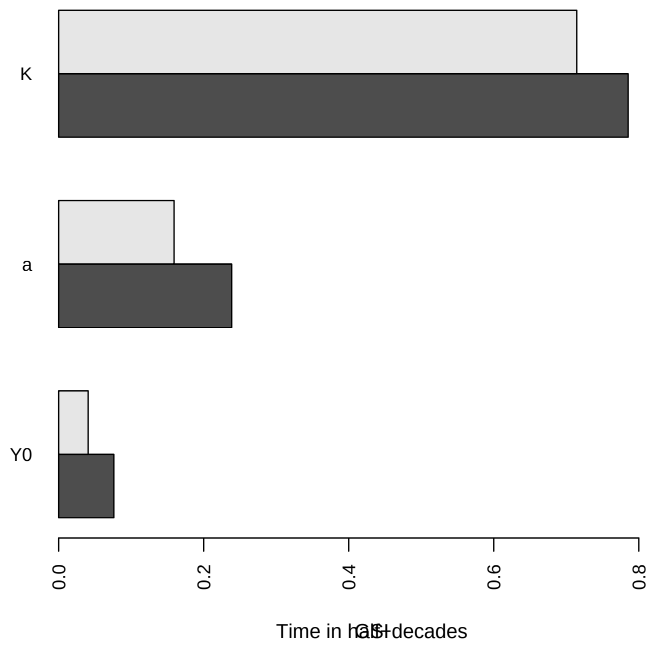


Figure 9: Evolution of the Fast sensitivity indices of the Verhulst model from $t = 5$ to $t = 100$. The upper subplot shows the extreme (tirets), inter-quartile (grey) and median (bold line) output values at all time steps. The lower subplot represents the sensitivity indices at all time steps for the main effects and the first-order interactions.

Acknowledgements

This document was typed using the `knitr` package ([12, 13]).

References

[1] K. Campbell, M. McKay & B. Williams – "Sensitivity analysis when model outputs are functions", *Reliability Enineering and Systems Safety* **91** (2006), p. 1468–1472.

[2] R. Faivre, B. Iooss, S. Mahévas, D. Makowski & H. Monod (éds.) – *Analyse de sensibilité et exploration de modèles. applications aux sciences de la nature et de l'environnement*, Collection Savoir-Faire. Éditions Quae, 2013.

[3] G. James, D. Witten, T. Hastie & R. Tibshirani (éds.) – *An introduction to statistical learning with applications in R*, Springer Texts in Statistics, Springer, 2013.

[4] M. Lamboni, D. Makowski, S. Lehuger, B. Gabrielle & H. Monod – "Multivariate global sensitivity analysis for dynamic crop models", *Field Crops Research* **113** (2009), p. 312–320.

[5] M. Lamboni, H. Monod & D. Makowski – "Multivariate sensitivity analysis to measure global contribution of input factors in dynamic models", *Reliability Engineering & System Safety* **96** (2011), p. 450–459.

[6] A. Lurette, S. Touzeau, M. Lamboni & H. Monod – "Sensitivity analysis to identify key parameters influencing *Salmonella* infection dynamics in a pig batch", *Journal of Theoretical Biology* **258** (2009), no. 1, p. 43–52.

[7] G. Pujol, B. Iooss & A. Janon – "Package 'sensitivity'. a collection of functions for factor screening, global sensitivity analysis and reliability sensitivity analysis of model output", Tech. Report Version 1.11.1, 2015, with contributions from Sebastien Da Veiga, Jana Fruth, Laurent Gilquin, Joseph Guillaume, Loic Le Gratiet, Paul Lemaitre, Bernardo Ramos, Taieb Touati.

[8] A. Redd – "A comment on the orthogonalization of B-spline basis functions and their derivatives", *Statistics and Computing* **22** (2011), no. 1, p. 251–257.

[9] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana & S.Tarantola – *Global sensitivity analysis: the primer*, Wiley, New York, 2008.

[10] S. Wood – *Generalized additive models: An introduction with R*, Chapman and Hall/CRC, 2006.

[11] H. Xiao & L. Li – "Discussion of paper by Matieyendou Lamboni, Hervé Monod, David Makowski "Multivariate sensitivity analysis to measure global contribution of input factors in dynamic models", Reliab. Eng. Syst. Saf. 99 (2011) 450-459", *Reliability Engineering & System Safety* **147** (2016), p. 194–195.

[12] Y. Xie – *Dynamic documents with R and knitr, 2nd edition.*, Chapman and Hall/CRC, 2015.

[13] — , "knitr: A general-purpose package for dynamic report generation in R", Tech. Report R package version 1.12, 2016.