Journal of Statistical Software

August 2013, Volume 54, Issue 9. *http://www.jstatsoft.org/*

Simultaneous Optimization of Multiple Responses with the R Package JOP

Sonja Kuhnt
TU Dortmund University

Nikolaus Rudak
TU Dortmund University

Abstract

A joint optimization plot, shortly JOP, graphically displays the result of a loss function based robust parameter design for multiple responses. Different importance of reaching a target value can be assigned to the individual responses by weights. The JOP method simultaneously runs through a whole range of possible weights. For each weight matrix a parameter setting is derived which minimizes the estimated expected loss. The joint optimization plot displays these settings together with corresponding expected values and standard deviations of the response variable. The R package **JOP** provides all tools necessary to apply the JOP approach to a given data set. It also returns parameter settings for a desirable compromise of achieved expected responses chosen from the plot.

Keywords: multiple responses, simultaneous optimization, Pareto optimality.

1. Introduction

In many technical applications, as in thermal spraying processes, it is desirable to find a setting of controllable machine parameters that brings the mean of multiple responses on target while simultaneously minimizing the variance. Usually these responses are inconsistent with one another and it is not possible to optimize all means and variances at the same time. Current solutions based on response surface methodology (Khuri and Mukhopadhyay 2010) are extensions of the desirability functions approach (Derringer and Suich 1980; Wu 2009; Köksoy 2005; He, Wang, Oh, and Park 2010) or the squared error loss approach (Shen, Zhao, and Yang 2010). The R packages **desire** (Trautmann, Steuer, and Mersmann 2012), **desirability** (Kuhn 2012) and **qualityTools** (Roth 2013) make use of the desirability function in order to perform multi response optimization. The R package **rsm** (Lenth 2009) provides several functions in order to apply response surface methods.

Pignatiello (1993) and Vining (1998) are beyond the first who extended the loss function approach to multiple responses. This extension involves a pre-specified and in general unknown cost matrix which can only be assumed on the basis of the underlying process. Thus it might be insufficient to only consider one cost matrix. Kuhnt and Erdbrügge (2004) introduce an alternative methodology where they minimize the estimated expected loss, namely the risk, for a whole sequence of cost matrices. For each cost matrix an optimal parameter setting together with corresponding responses is derived and graphically displayed by the so called joint optimization plot. This methodology is implemented in the R package **JOP** (Kuhnt and Rudak 2013; R Core Team 2013).

Our article is organized as follows. First we give a brief introduction to the joint optimization plot (JOP) methodology. Afterwards we present the implementation of the JOP method in R. We explain the main function `JOP` which performs the multi response optimization and produces the joint optimization plot. We demonstrate its use on the basis of a data set stored in the R package **JOP**. Afterwards we shortly explain the generation of the graphical output by `plot.JOP` and the usage of the auxiliary function `locate`. The function `locate` helps to find a "good" compromise based on the output of `JOP`. We close the article with a real data example.

2. The JOP method

Let us consider a production process with controllable machine parameters $x_1, \ldots, x_n$ and p independent quality characteristics, represented by variables $Y = Y_1, \ldots, Y_p$, with target values $\tau = \tau_1, \ldots, \tau_p$. The aim is a layout of the production process, in terms of choosing values for $x = (x_1, \ldots, x_n)$, which produces outcomes of the quality characteristics always on or very close to the target values. In other words, parameter values are searched to ensure that the means of the quality characteristics are on target with minimal variances. This task can be achieved by minimizing the risk function (Pignatiello 1993), that is the expected loss, over x,

$$\begin{aligned} R(x) &= \mathsf{E}\big(\text{loss}(Y|x)\big) = \mathsf{E}((Y-\tau)^\top C(Y-\tau)|x) \\ &= \text{trace}\big(C\Sigma(x)\big) + \big(\mu(x)-\tau\big)^\top C\big(\mu(x)-\tau\big) \end{aligned} \tag{1}$$

where $\text{loss}(Y) = (Y-\tau)^\top C(Y-\tau)$ is the loss function, C the positive definite cost matrix, $\mu(x) = \mathsf{E}(Y \mid x)$ the expected value of Y given x, and $\Sigma(x) = \mathsf{COV}(Y \mid x)$ the covariance matrix of Y given x. The cost matrix C can be derived from knowledge of occurring losses for specific outcomes of the quality characteristics. Often such information is not available, in which case the joint optimization approach provides a solution as will be seen below.

For independent quality characteristics $Y_1, \ldots, Y_p$, the covariance matrix $\Sigma(x)$ is diagonal and C might also be chosen diagonal, such that Equation 1 reduces to

$$R(x) = \sum_{r=1}^{p} c_r \cdot (\sigma_r^2(x) + (\mu_r(x) - \tau_r)^2) = \sum_{r=1}^{p} c_r \cdot g_r(x) \tag{2}$$

where c_r is the r-th diagonal entry of the cost matrix C and $g_r(x) = \sigma_r^2(x) + (\mu_r(x) - \tau_r)^2$. Equation 2 points out that minimizing the risk function brings the mean on target and minimizes the variances.

A point $y^* \in f(\mathcal{R})$ of a vector-valued function $f : \mathcal{R} \subset \mathbb{R}^n \to f(\mathcal{R}) \subset \mathbb{R}^k$ is called efficient with regard to the order relation $\leq$ defined in $\mathbb{R}^n$, if and only if there exists no other $y \in f(\mathcal{R})$, $y \neq y^*$, with $y \leq y^*$. A point $x^* \in \mathbb{R}^n$ with $y^* = f(x^*)$ is called *Pareto optimal*, if and only if y^* is efficient (Erdbrügge, Kuhnt, and Rudak 2011; Hillermeier 2001). Erdbrügge *et al.* (2011) show that the optimal point x^* that minimizes the risk is also Pareto optimal for the vector valued optimization problem $\min_{x \in X \subset \mathcal{R}^n} (g_1(x), \ldots, g_p(x))^\top$.

The mean vector $\mu(x)$ and the covariance matrix $\Sigma(x)$ are unknown and need to be estimated. As a general class of models double generalized linear models (Aitkin 1987; McCullagh and Nelder 1989; Smyth 1989; Engel and Huele 1996; Smyth and Verbyla 1999) allow to derive estimated mean and variance models for different data situations. Often double generalized linear models with identity link and normal probability assumption for the mean model need to be fitted, hence

$$\widehat{\mathsf{E}}(Y_r|x) = f(x). \tag{3}$$

The variance model is based on the squared residuals of the mean model as Gamma distributed responses and the log link resulting in

$$\widehat{\mathsf{VAR}}(Y_r|x) = \exp\{g(x)\}. \tag{4}$$

However, this is only an example of the wide range of possible model specifications out of the class of double generalized linear models. In any case, f and g are both functions in the unknown parameters. An iterative fitting procedure for both models (3) and (4) alternates between fitting the mean and variance model, at each step using the actual estimates.

In situations with an unknown matrix C the joint optimization plot approach can be applied, as it considers a whole sequence of possible cost matrices C simultaneously. By means of the joint optimization plot (Kuhnt and Erdbrügge 2004; Erdbrügge *et al.* 2011) the user can choose a compromise based on the knowledge of the underlying process. The cost matrix C is decomposed into

$$C = A^\top W A, \tag{5}$$

where A is a so called standardization matrix and W a weight matrix. We take both A and W to be diagonal matrices. The diagonal entries of the weight matrix W indicate the importance of the corresponding response; these diagonal entries are further specified by a slope vector $d \in \mathcal{R}^p$ and a stretch value $\log a$ in the following way

$$\log w = d \cdot \log a \tag{6}$$

where w is the diagonal of the weight matrix W.

The standardization matrix A ensures that the loss function and thereby the minimized risk is invariant to transformations of the individual responses. Implemented in **JOP** is the standardization matrix

$$A_Y = \operatorname{diag}\left(\left[\frac{1}{m}\sum_{i=1}^{m}\widehat{\mathsf{VAR}}(Y_r|x^i)\right]^{-1/2}_{r=1,\ldots,p}\right),$$

for which the risk is invariant to affine linear transformations. Thus the risk is the same for different units, like cm or mm for example.

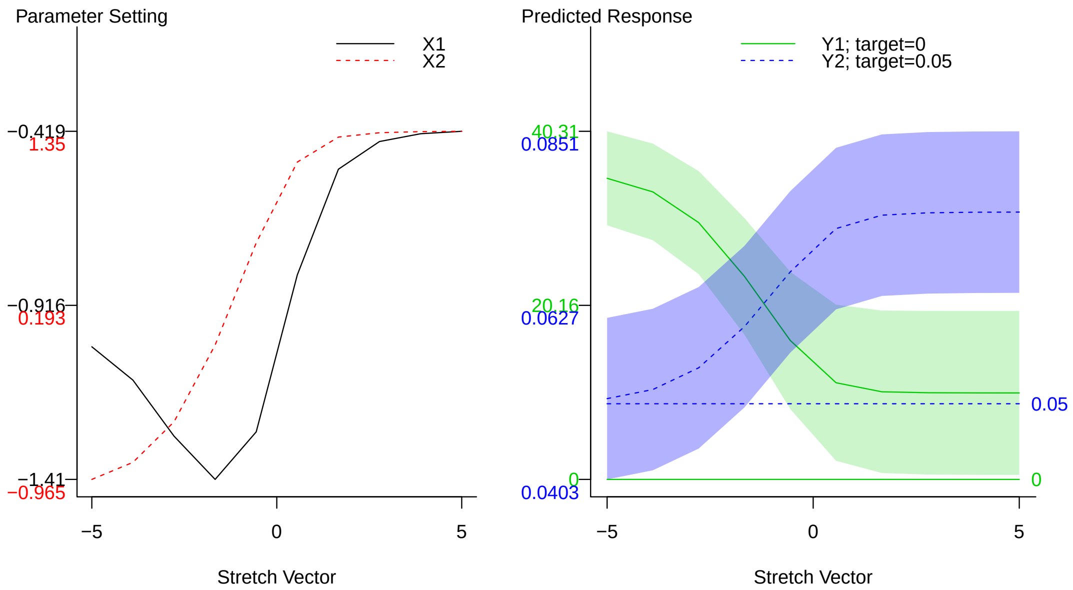

Figure 1: Examplary joint optimization plot (JOP).

For diagonal standardization and weight matrices the estimated risk function in Equation 2 reduces to

$$\hat{R}(x) = \sum_{r=1}^{p} w_r \cdot \frac{(\widehat{\mathsf{VAR}}(Y_r|x) + (\widehat{\mathsf{E}}(Y_r|x) - \tau_r)^2)}{b_r^2} \tag{7}$$

where

$$b_r = ([A]_{rr})^{-1} = \left(\frac{1}{m}\sum_{i=1}^{m}\widehat{\mathsf{VAR}}(Y_r|x^i)\right)^{\frac{1}{2}}$$

denotes the inverse of the $[r,r]$-entry of the standardization matrix A.

The risk function (7) is not only minimized with respect to x for an individual cost matrix, but moreover for a whole sequence of cost matrices C_t. A slope vector $d \in \mathcal{R}^p$ is chosen together with a vector $\log a \in \mathcal{R}^T$ of equidistant stretch values that range between a minimal value *Wstart* and a maximal *Wend*, namely $\log a = [Wstart, \ldots, Wend]$, resulting in $C_t = A^\top W_t A$ with $W_t = \exp(d \cdot \log a_t)$, $t = 1\ldots,T$. The joint optimization plot displays the optimal parameters $\hat{x}_t$ for every C_t in one plot and the corresponding predicted mean values $\widehat{\mathsf{E}}(Y_r|\hat{x}_t)$ for $r = 1,\ldots,p$ together with a band of width twice the standard deviation $\sqrt{\widehat{\mathsf{VAR}}(Y_r|\hat{x}_t)}$ in a second plot, as illustrated in Figure 1. The horizontal dashed lines on the right hand plot stand for the target values to be reached. The calculated optimal parameters on the left hand plot and corresponding responses on the right hand plot are interpolated to enable a better understanding of the results. Now the user can choose a compromise on the right hand plot and find the corresponding design parameter values on the left hand plot.

For the sake of readability of the joint optimization plot we recommend to deal with at most five responses. The number of machine parameters might as well be higher. However, a higher number of machine parameters and responses leads to an increase of computation time.

3. Implementation in R

The core of the JOP method is the minimization of the risk function (7) for a prespecified sequence of cost matrices. The **JOP** package comprises the following components:

- `JOP`: Automated model building by means of the function `dglm` out of the R package **dglm** (Dunn and Smyth 2012), and optimization of the risk function by `solnp` out of the R package **Rsolnp** (Ghalanos and Theussl 2012).
- `plot.JOP`: Visualization.
- `locate`: Selection of a "good" compromise by mouse click.
- `datax` and `datay`: Data sets, `datax` contains the experimental design with two parameter settings for a sheet metal hydroforming process and `datay` includes the corresponding experimental results for two responses (Kuhnt and Erdbrügge 2004).

In this section we present the general call of `JOP` and explain the input arguments and output values in more detail. Furthermore, we demonstrate the usage of `JOP` based on the data sets `datax` and `datay`.

3.1. Structure of `JOP`

The general call of `JOP` is as follows.

```
JOP(datax, datay, tau = "min", Wstart = -5, Wend = 5, numbW = 10,
  d = NULL, optreg = "sphere", Domain = NULL, form.mean = NULL,
  form.disp = NULL, family.mean = NULL, dlink= "log", mean.model = NULL,
  var.model = NULL, joplot = FALSE, solver = "solnp")
```

In the following we describe the parameters used in the function `JOP`.

- `Wstart`, `Wend`, `numbW`, `d`: These parameters assign the sequence of weight matrices W, compare (6), in the following way:

$$W_t = \operatorname{diag}\left(\exp\left(d \cdot (Wstart + t \cdot \frac{Wend - Wstart}{numbW})\right)\right), \text{ for } t \in 0, \ldots, numbW \quad (8)$$

- `optreg`: The optimization region is specified, `optreg = "box"` for box constraints or `optreg = "sphere"` (default) for sphere region.
- `Domain`: Optional argument for the specification of box constraints for each response, lower in the first column and upper constraints in the second.
- `tau`: A `list` object or a single character value which specifies the target values for the responses. The target values can be either numerical or characters (`"min"` for minimization or `"max"` for maximization). If a target is specified by `"min"` or `"max"` then `JOP` derives the possible minimal or maximal values based on the fitted models and optimization region and takes these values as target values internally.

On the one hand, the user can plug in a list of numerical values or characters `"min"` for minimization or `"max"` for maximization as targets for the corresponding response.

On the other hand, the user can either set `tau = "max"` or `tau = "min"` in order to maximize or minimize all responses. By default (`tau = "min"`) all responses are minimized.

- `solver`: The optimization is performed by `solnp` by default. The user can also choose `solver = "gosolnp"`. Especially when a function is higly complex and has perhaps many local minima, it is recommended to use `gosolnp`.
- `form.mean`, `form.disp`: A list of `formulas` for the mean and dispersion of each response.
- `mean.model`, `var.model`: Lists of functions for the mean and variance of each response.
- `family.mean`: Family object, distribution assumption and link specification for the mean and dispersion.
- `dlink`: List of names of link functions for each dispersion model of each response.
- `datax`: Data frame which contains an experimental design.
- `datay`: Data frame which contains responses.

The data sets `datax` and `datay` are needed for model building. Both `datax` and `datay` have to be data frames where `datax` contains an experimental design with settings for each parameter columnwise and `datay` contains the experimental results columnwise for every response. Additionally, the columns of the data sets should be named, as exemplary demonstrated by the data sets contained in the package **JOP**.

Based on the lists of formulas and the distribution assumptions `JOP` builds double generalized linear models for each response. First, a model for the mean is fitted with constant dispersion model. Afterwards a combination of forward and backward selection is performed for the mean model. Then a double generalized linear model is fitted with the mean model consisting of main effects, interaction effects and quadratic effects and the dispersion model including all main effects. Thereafter, a backward selection for each dispersion model is performed dropping the least significant covariate in each step. In a final step `JOP` checks if for each higher order effect the corresponding main effect is included in the model. If not, then the corresponding main effect is added to the model and the double generalized linear model is fitted again.

The arguments are summarized in Table 1. `JOP` returns an object of class "`JOP`". The values stored in the output list are summarized in Table 2. In addition to the optimal settings of the parameters for each weight matrix together with the corresponding responses the output contains the estimated standard deviations for each response and minimal risk function values. Furthermore the double generalized linear models are stored. Moreover `JOP` returns the parameters together with the corresponding responses that minimize the sum of single risk functions among all calculated parameters. In order to reconstruct the calculations the output also contains the input variables `d`, `Wstart`, `Wend` and `numbW`.

Argument	Description
General	
`datax`	Data set with experimental design
`datay`	Data set with responses
`joplot`	Graphical output if `joplot = TRUE`
Sequence of weight matrices	
`Wstart`, `Wend`, `numbW`, `d`	See (8)
Models for mean and dispersion	
`form.mean`	List of formulas for the mean of each response
`form.disp`	List of formulas for the dispersion of each response
`family.mean`	Family object for the mean
`dlink`	List of names of link functions for each dispersion model of each response
`mean.model`	List of functions for the mean of each response
`var.model`	List of functions for the variance of each response
Optimization	
`optreg`	Specifies optimization region (`"box"` for box optimization constraints and `"sphere"` for sphere)
`solver`	`"solnp"` or `"gosolnp"`
`tau`	Vector of target values
`Domain`	Box constraints for each response

Table 1: Summary of input arguments for main function `JOP`.

Output value	Description
`Parameters`	Optimal settings of input parameters
`Responses`	Corresponding predicted mean responses
`StandardDeviation`	Corresponding predicted standard deviations
`OptimalValue`	Minimal risk function value
`TargetValueJOP`	Target values used internally by `JOP`
`TargetValueUSER`	Target values specified by the user
`DGLM`	Stored models for mean and dispersion
`RiskminimalParameters`	Selected parameter settings which minimize the sum of squared single risk functions among all parameters
`RiskminimalResponses`	Responses associated with `RiskminimalParameters`
`valW`	Values for `Wend` and `Wstart`, see Table 1
`d`	Slope vector, see (6)
`numbW`	Number of weight matrices, see Table 1

Table 2: Summary of output values for main function `JOP`.

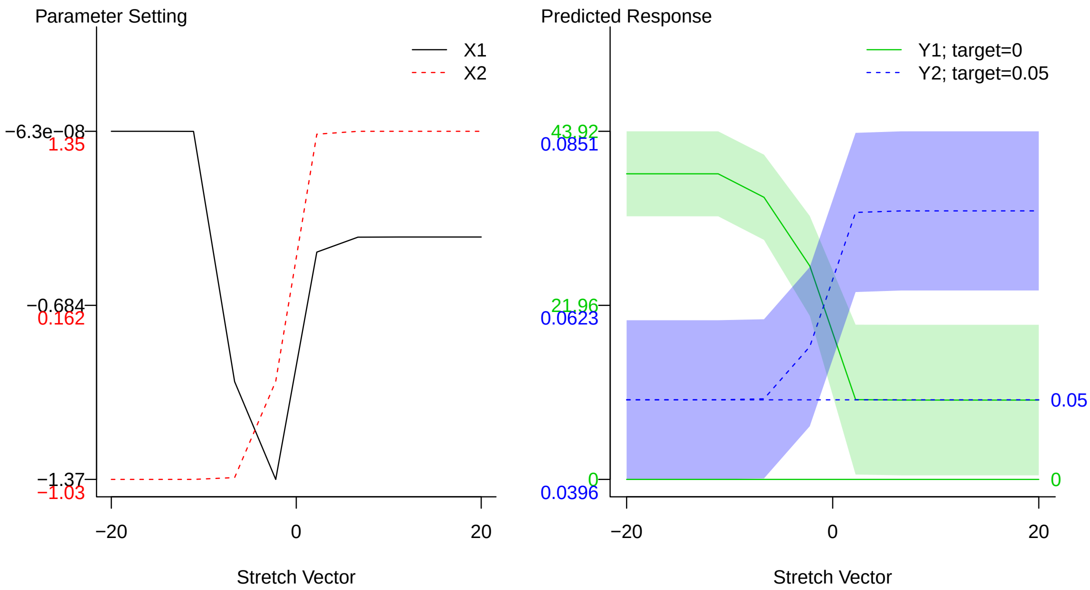

Figure 2: Joint optimization plot generated by `JOP` with `Wend = -20` and `Wstart = 20`.

3.2. Exemplary data set

We demonstrate the usage of `JOP` step by step based on the exemplary data sets `datax` and `datay` (Table 6 and Table 7 in the Appendix A) stored in the R package **JOP** which are loaded automatically. The data sets come from an experiment with two input parameters (`datax`), X_1 and X_2, and two output variables (`datay`), Y_1 and Y_2, with 36 runs in total. The values `Wend` and `Wstart` can be chosen based on experience, otherwise the user should start with relatively large values, for example a sequence of 10 weight matrices (`numbW = 10`, which is default value and need not to be specified) with a stretch vector ranging between -20 (`Wstart = -20`) and 20 (`Wend = 20`). The slope vector is (1,0) in this example (`d = c(1, 0)`). We want `JOP` to build double generalized linear models for the mean and dispersion for Y_1 and Y_2. The target values are given by 0 and 0.05 (`tau = list(0 , 0.05)`). This leads to the following call which generates Figure 2.

```
R> out1 <- JOP(datax = datax, datay = datay, tau = list(0, 0.05),
+    Wstart = -20, Wend = 20, joplot = TRUE)
```

In Figure 2 it becomes clear that the lines in both plots are mostly constant except the part in the center of the plot. We therefore reduce the range of the stretch vector by setting `Wend = -5` and `Wstart = 5` (default values, thus need not to be specified).

```
R> out2 <- JOP(datax = datax, datay = datay, tau = list(0, 0.05),
+    joplot = TRUE)
```

This gives Figure 3. It can be observed that the band width in the right hand plot varies due to the variance model depending on the parameters. The following double generalized linear model has been build by `JOP` for the response Y_1.

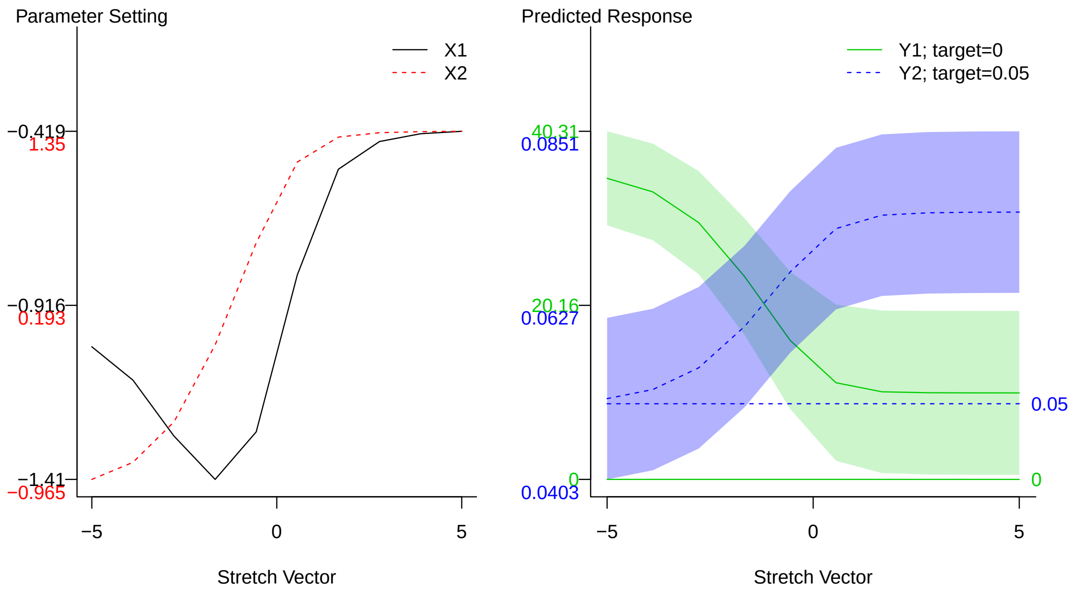

Figure 3: Joint optimization plot generated by `JOP` with `Wend = -5` and `Wstart = 5`.

```
R> summary(out2$DGLM$Y1)

Call: dglm(formula = flist[[i]], dformula = dispf[[i]], family =
    family.mean[[i]], dlink = dlink, data = dataset, method = "reml")

Mean Coefficients:
              Estimate Std. Error   t value     Pr(>|t|)
(Intercept)  26.740494   1.196697 22.345247 1.683592e-21
X1            2.866961   1.178760  2.432183 2.059735e-02
X2          -11.493717   1.231388 -9.333952 8.835328e-11
(Dispersion Parameters for gaussian family estimated as below )

    Scaled Null Deviance: 125.5757 on 35 degrees of freedom
Scaled Residual Deviance: 32.8789 on 33 degrees of freedom

Dispersion Coefficients:
             Estimate Std. Error   z value     Pr(>|z|)
(Intercept) 3.8527027  0.2571879 14.980108 9.905641e-51
X2          0.4805173  0.2752690  1.745628 8.087558e-02
(Dispersion parameter for Gamma family taken to be 2 )

    Scaled Null Deviance: 48.10039 on 35 degrees of freedom
Scaled Residual Deviance: 45.07052 on 34 degrees of freedom

Minus Twice the Log-Likelihood: 237.7398
Number of Alternating Iterations: 7
```

As it can be seen in Figure 3, either the target of Y_1 or Y_2 can be nearly reached but not both at the same time. Hence, a decision is needed which response should be preferred or a compromise has to be set in the sense that both responses are kept as small as possible. The calculated parameters and corresponding responses are as follows.

```
R> out2$Responses

          Y1         Y2
W1  34.86805 0.05065696
W2  33.29256 0.05183249
W3  29.73682 0.05462793
W4  23.50584 0.05993138
W5  16.08850 0.06697651
W6  11.19272 0.07255414
W7  10.14573 0.07427949
W8  10.03864 0.07458055
W9  10.01937 0.07465548
W10 10.01432 0.07467799
```

```
R> out2$Parameters

            X1          X2
W1  -1.0338141 -0.96500174
W2  -1.1290056 -0.85167271
W3  -1.2888263 -0.58217417
W4  -1.4124358 -0.07088651
W5  -1.2766983  0.60831034
W6  -0.8286217  1.14603052
W7  -0.5269680  1.31236607
W8  -0.4479661  1.34138970
W9  -0.4257229  1.34861411
W10 -0.4187866  1.35078414
```

We can use the function `locate` in order select a compromise by means of clicking with the mouse on the right hand plot (see Figure 4). Afterwards the chosen responses are returned along with the corresponding parameters .

```
R> locate(out2 , xlu = 4)

$ChosenParameters
         X1          X2
-1.41243583 -0.07088651

$ChosenResponses
         Y1          Y2
23.50584426  0.05993138
```

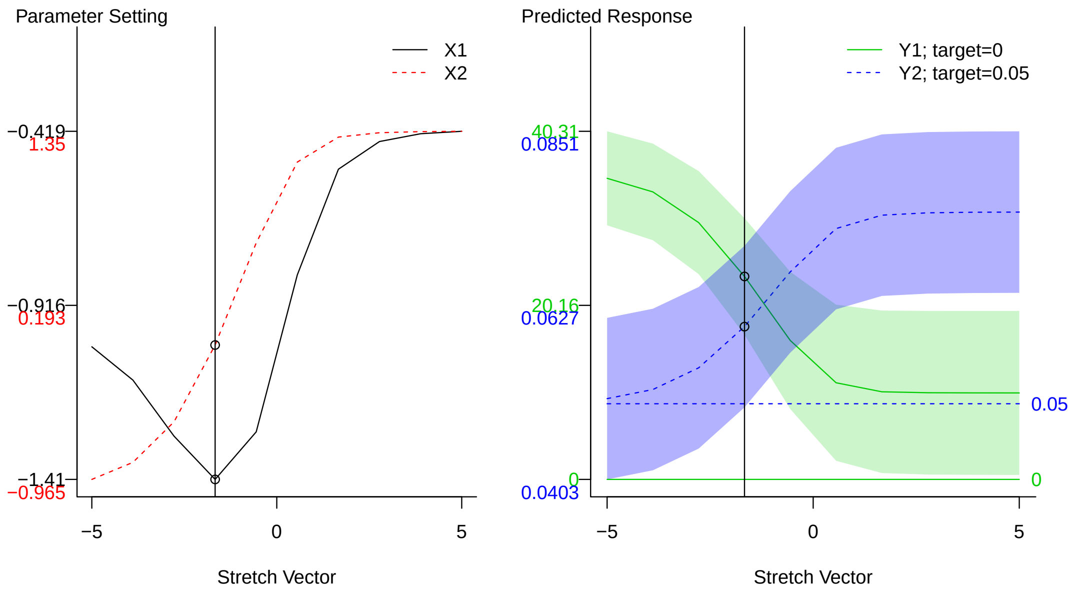

Figure 4: Compromise found by means of `locate`.

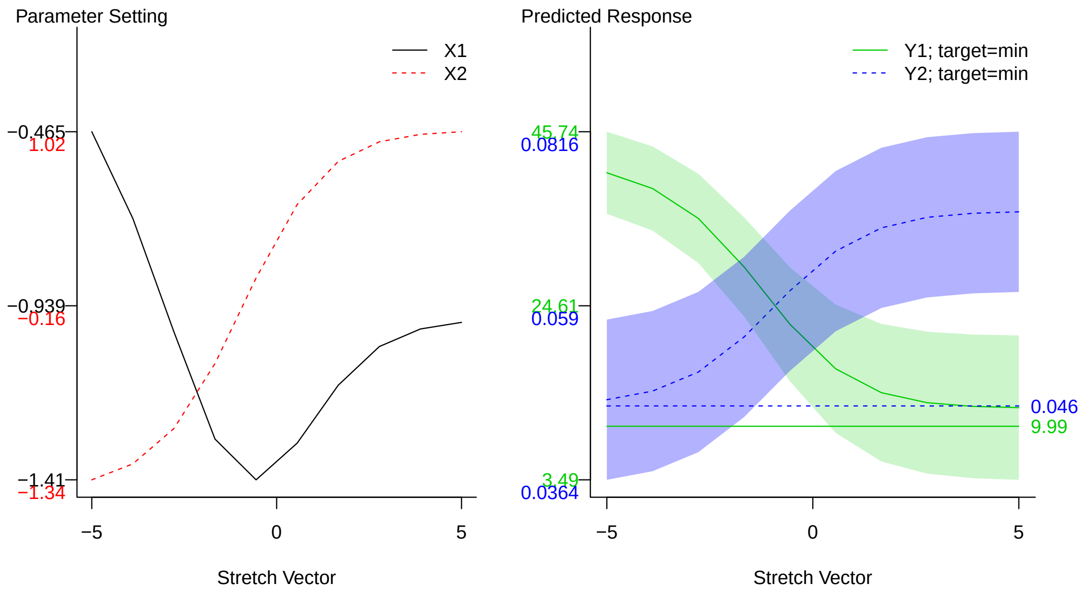

Figure 5: Joint optimization plot with targets `"min"` for Y_1 and `"min"` for Y_2.

Alternatively, the user can set the target values to minimum (Default). Figure 5 contains the plot generated by the following code.

```
R> out3 <- JOP(datax = datax, datay = datay, joplot = TRUE)
```

The user can also specify formulas for the mean and dispersion of each response as demonstrated next. The corresponding joint optimization plot is displayed in Figure 6.

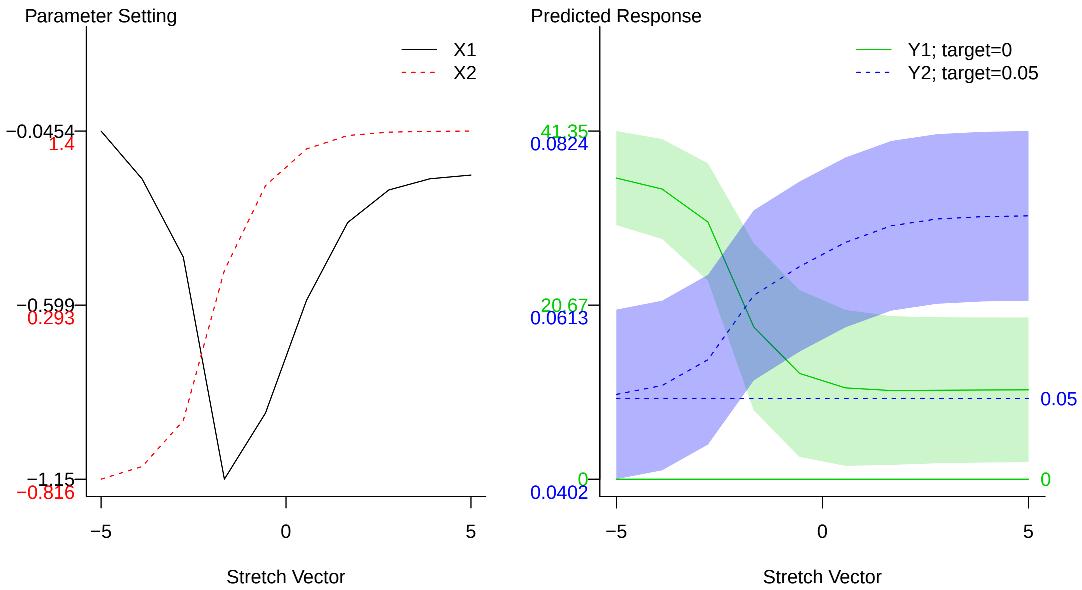

Figure 6: Joint optimization plot for output `out4`.

```
R> form.mean <- list(as.formula(Y1 ~ X1 + X2 + I(X1^2)),
+    as.formula(Y2 ~ (X1 + X2)^2 + I(X1^2)))
R> form.disp <- list(as.formula(d ~ X1 + X2 + I(X2^2)), as.formula(d ~ X1))
R> out4 <- JOP(datax = datax, datay = datay, tau = list(0, 0.05),
+    form.mean = form.mean, form.disp = form.disp, joplot = TRUE)
```

Another possibility is to plug in lists of functions for the mean and dispersion as shown by the following code. The graphical output is displayed in Figure 7.

```
R> mean1 <- function(x) {
+    return(26.917 + 2.797 * x[1] - 11.104 * x[2])
+  }
R> mean2 <- function(x) {
+    return(0.063649 - 0.00179 * x[1] + 0.010198 * x[2] -
+      0.003346 * x[2]^2 + 0.00401 * x[1] * x[2])
+  }
R> var1 <- function(x) {
+    return(exp(3.7851 - 0.3621 * x[1] + 0.536 * x[2]))
+  }
R> var2 <- function(x) {
+    return(exp(-10.51784 - 0.01324 * x[1] + 0.5147 * x[2] +
+      0.58101 * x[1]^2 + 0.82336 * x[2]^2))
+ }
R> mean.model <- list(mean1, mean2)
R> var.model <- list(var1, var2)
```

These functions are then handed over to the JOP procedure.

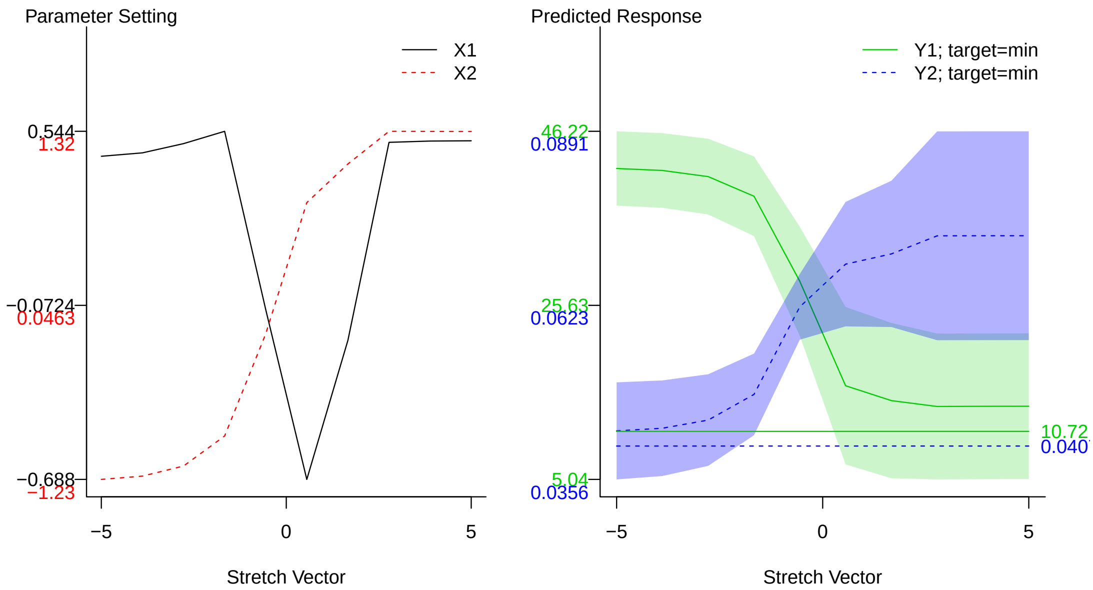

Figure 7: Joint optimization plot for output `out5`.

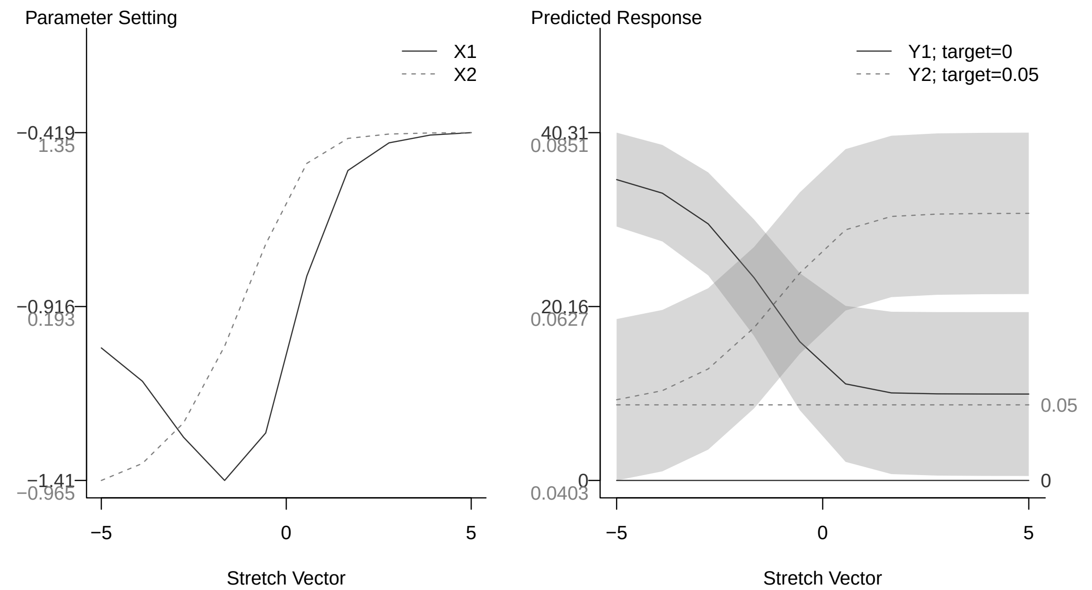

Figure 8: Grey scaled.

```
R> out5 <- JOP(datax = datax, datay = datay, mean.model = mean.model,
+    var.model = var.model, joplot = TRUE)
```

Furthermore we can generate a grey scaled joint optimization plot (see Figure 8).

```
R> plot(out2, no.col = TRUE)
```

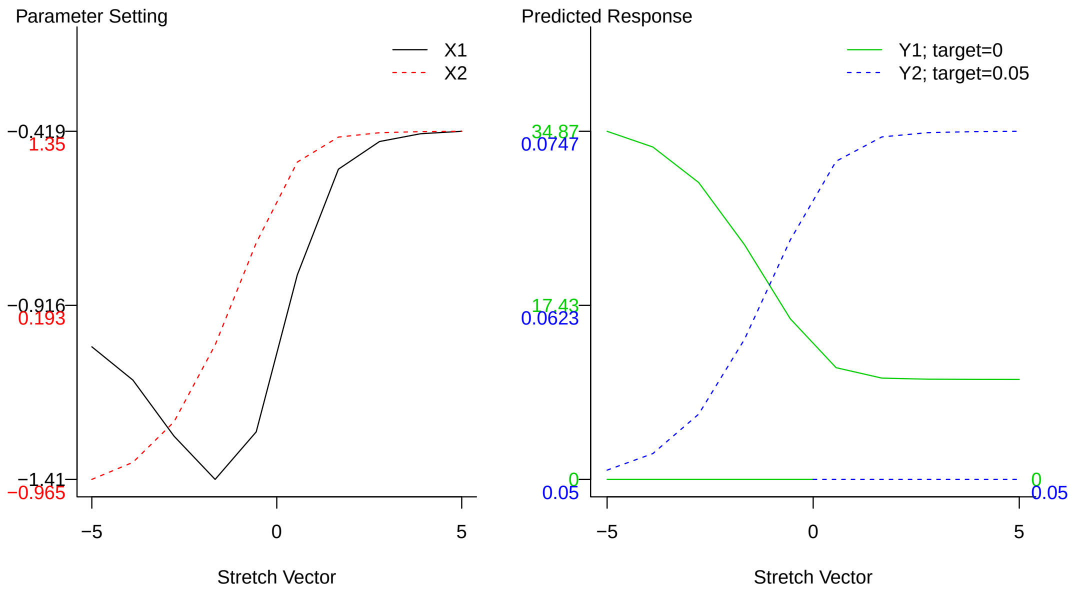

Figure 9: Without band.

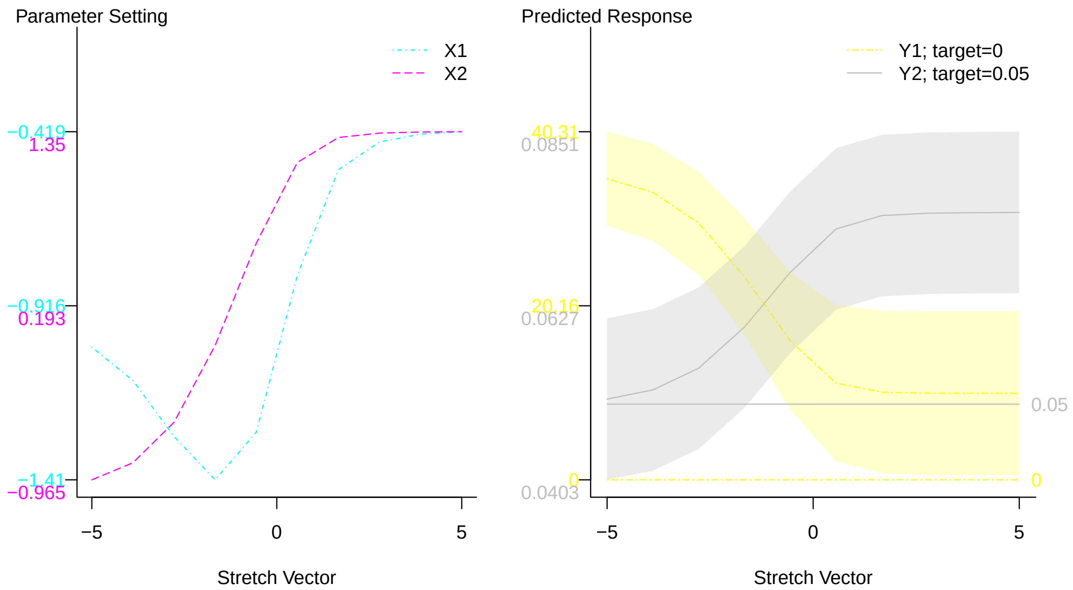

Figure 10: Optional `col` and `lty`.

Sometimes it is helpful to leave out the bands in the right hand plot to get the general idea (see Figure 9). This can be done by the following call.

```
R> plot(out2, standard = FALSE)
```

Additionally, several graphical parameters can be passed to `plot.JOP`. The arguments `col` or `lty` should have the same length as the number of parameters plus the number of responses.

Otherwise `plot.JOP` takes the default colors defined by the argument `no.col`. Figure 10 can be generated by the following code.

```
R> plot(out2, col = 5:8, lty = 4:7)
```

4. Real data example

In this example we analyze a thermal spraying process. The thermal spraying technology can be used to apply a particle coating on a surface, e.g., for wear protection or durable medical instruments. Thermal spraying processes are lacking in reproducibility due to uncontrollable day-effects. Furthermore, the analysis of the quality of the coating is very time-consuming. Therefore, a recent project studies in-flight particle properties which can be measured online. The uncontrollable day effects are expected to be observed through the particle properties which have a high impact on the coating. We aim to control the process through online-diagnosis of the in-flight particles. Hence, we model the relationship between the controllable machine parameters and in-flight particle properties. The design consists of 30 runs in total. There are four different controllable machine parameters with five different chosen settings, summarized in Table 3, and two different in-flight particle properties, namely the velocity and the temperature. The experimental set up is visualized in Figure 11. In order to model the relationship between the controllable machine parameters and the in-flight particles, a central composite design was performed.

The data set can be found in Table 8 in the Appendix A. In the following `sprayX` contains the experimental design and `sprayY` contains the experimental results for velocity and temperature. Now we can call `JOP` in order to build models with main effects, interactions and quadratic effects and to get the joint optimization plot. The following code

```
R> out <- JOP(datax = sprayX, datay = sprayY, tau = list(1550, 750),
+     optreg = "box", joplot = TRUE)
```

calls the main function `JOP`. We use the slope vector `d = c(1, 0)` and take a stretch vector with `numbW = 10` equidistant values between `Wstart = -5` and `Wend = 5` (default values).

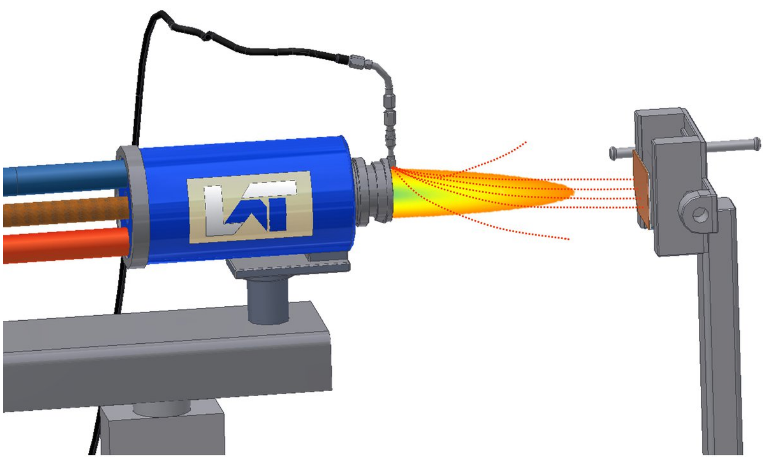

Figure 11: Experimental setup.

	Level				
Factor	-2	-1	0	1	2
Lambda (L)	1.0	1.075	1.15	1.225	1.3
Kerosene level (K) in $\frac{l}{h}$	15	17.5	20	22.5	25
Stand-off distance (D) in mm	200	225	250	275	300
Feeder Disc Velocity (FDV) in %	5	10	15	20	25

Table 3: Parameter values.

Furthermore we set the target value `tau = list(750, 1550)` for the velocity and the temperature. The distribution assumption is `gaussian(link = "identity")` for the mean and `Gamma(link = "log")` for the dispersion.

Now we take a look at the fitted selected models. The mean model for velocity includes all main effects and a quadratic term for kerosene level and an interaction between distance and feeder disc velocity. Temperature depends on the same parameters but with different signs for the coefficients of lambda and feeder disc velocity. Both dispersion models depend on distance.

```
R> summary(out$DGLM$Ve)

Call: dglm(formula = flist[[i]], dformula = dispf[[i]], family =
    family.mean[[i]], dlink = dlink, data = dataset, method = "reml")

Mean Coefficients:
              Estimate Std. Error    t value     Pr(>|t|)
(Intercept) 720.965261   1.756216 410.521949 5.872676e-46
L             9.169296   1.324358   6.923577 4.659843e-07
K            38.479212   1.324358  29.054989 1.239551e-19
D            -8.835360   1.372279  -6.438456 1.433925e-06
FDV          -5.600081   1.552577  -3.606960 1.484517e-03
I(K^2)       -4.107965   1.284199  -3.198855 3.989576e-03
D:FDV        -4.691714   1.832970  -2.559624 1.751975e-02
(Dispersion Parameters for gaussian family estimated as below )

    Scaled Null Deviance: 896.3855 on 29 degrees of freedom
Scaled Residual Deviance: 20.91231 on 23 degrees of freedom

Dispersion Coefficients:
             Estimate Std. Error   z value     Pr(>|z|)
(Intercept) 4.0003764  0.3384877 11.818379 3.136804e-32
D           0.7187822  0.3943383  1.822755 6.834046e-02
(Dispersion parameter for Gamma family taken to be 2 )

    Scaled Null Deviance: 61.66351 on 29 degrees of freedom
Scaled Residual Deviance: 54.86265 on 28 degrees of freedom
```

```
Minus Twice the Log-Likelihood: 196.0599
Number of Alternating Iterations: 6
```

Here is the fitted selected model for the temperature.

```
R> summary(out$DGLM$Te)

Call: dglm(formula = flist[[i]], dformula = dispf[[i]], family =
    family.mean[[i]], dlink = dlink, data = dataset, method = "reml")

Mean Coefficients:
               Estimate Std. Error    t value     Pr(>|t|)
(Intercept) 1622.811760   2.706422 599.615162 9.657184e-50
L             -7.172888   2.034521  -3.525591 1.811393e-03
K             29.477723   2.034521  14.488780 4.719962e-13
D            -17.294695   2.109905  -8.196905 2.819053e-08
I(K^2)        -5.908166   1.975646  -2.990498 6.534414e-03
FDV            2.695564   2.393905   1.126011 2.717733e-01
D:FDV         -5.459576   2.824036  -1.933253 6.560727e-02
(Dispersion Parameters for gaussian family estimated as below )

    Scaled Null Deviance: 314.0419 on 29 degrees of freedom
Scaled Residual Deviance: 21.34625 on 23 degrees of freedom

Dispersion Coefficients:
             Estimate Std. Error   z value     Pr(>|z|)
(Intercept) 4.8424770  0.3386096 14.301063 2.154916e-46
D           0.7277245  0.3945376  1.844499 6.511038e-02
(Dispersion parameter for Gamma family taken to be 2 )

    Scaled Null Deviance: 40.18143 on 29 degrees of freedom
Scaled Residual Deviance: 33.87384 on 28 degrees of freedom

Minus Twice the Log-Likelihood: 221.7569
Number of Alternating Iterations: 14
```

In Figure 12 it can be observed that the predicted mean values for velocity nearly reach the desired target value on the left side of the plot. The temperature is maximal here. The opposite is true for the right side of the plot. Furthermore, the variances on left and right side of the plot are lower than in the middle. Thus we choose three different compromises, one in the middle, one on the left side and one on the right side, as exemplary illustrated in Figure 13. This can be done by

```
R> loc <- locate(out, ncom = 3)
```

and the chosen predicted response values and corresponding parameters are stored in Table 4 and Table 5. It can be seen from Table 4 that the solutions 2 and 3 are not as close to the

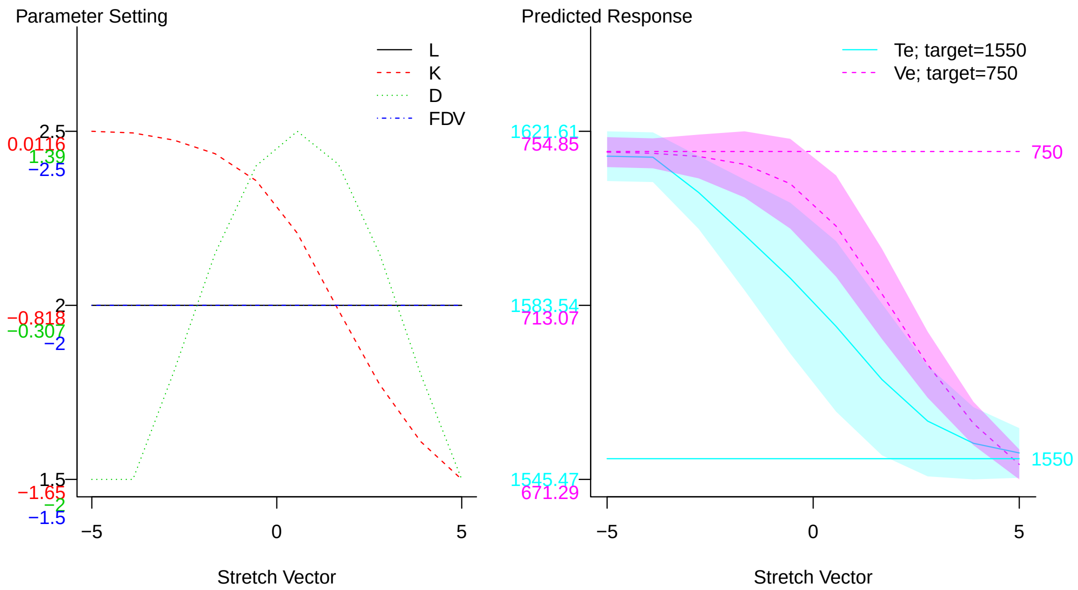

Figure 12: Joint optimization plot for thermal spraying process.

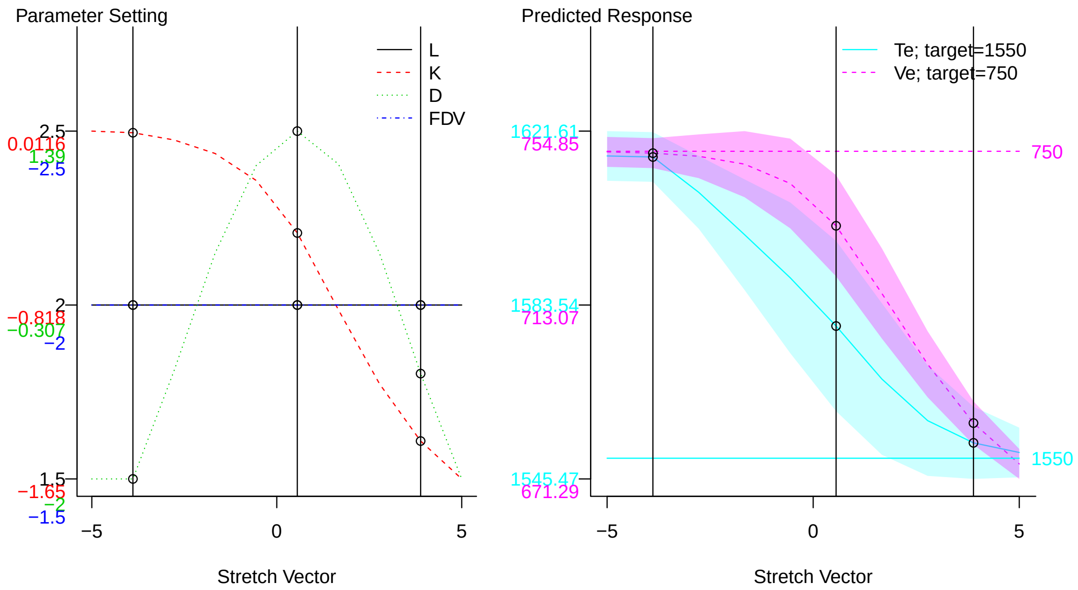

Figure 13: Three possibly chosen compromises for the thermal spraying process.

mean as solution 1 but show lower variance. Additionally, we calculated a desirability based solution (DES) in order to compare the results. We used two sided linear desirabilities as follows.

		Temperature		Velocity	
		Mean	Variance	Mean	Variance
JOP	Solution 1	1578.94	18.64	732.10	12.16
	Solution 2	1553.37	7.90	684.72	5.21
	Solution 3	1615.94	5.44	749.56	3.60
DES		1574.03	23.31	731.24	15.16

Table 4: Summarized solutions based on JOP method and on desirabilities.

		L	K	D	FDV
JOP	Solution 1	2.00	−0.47	1.39	−2.00
	Solution 2	2.00	0.004	−2.00	−2.00
	Solution 3	2.00	−1.47	−0.98	−2.00
DES		2.00	−0.50	2.00	−2.00

Table 5: Summarized parameters based on JOP method and on desirabilities.

$$\text{Temperature: } d_{te}(x) = \begin{cases} 0, & \hat{f}_{Te}(x) < 1500 \text{ or } \hat{f}_{Te}(x) > 1600 \\ \frac{\hat{f}_{Te}(x)-1500}{50}, & \hat{f}_{Te}(x) \geq 1500 \text{ and } \hat{f}_{Te}(x) < 1550 \\ \frac{\hat{f}_{Te}(x)-1550}{50}, & \hat{f}_{Te}(x) \geq 1550 \text{ and } \hat{f}_{Te}(x) < 1600 \end{cases}$$

$$\text{Velocity: } d_{ve}(x) = \begin{cases} 0, & \hat{f}_{Ve}(x) < 700 \text{ or } \hat{f}_{Ve}(x) > 800 \\ \frac{\hat{f}_{Ve}(x)-700}{50}, & \hat{f}_{Ve}(x) \geq 700 \text{ and } \hat{f}_{Ve}(x) < 750 \\ \frac{\hat{f}_{Ve}(x)-750}{50}, & \hat{f}_{Ve}(x) \geq 750 \text{ and } \hat{f}_{Ve}(x) < 800 \end{cases},$$

where $\hat{f}_{Te}(x)$ and $\hat{f}_{Ve}(x)$ denote the fitted mean models for temperature and velocity. Then the overall desirability index $d(x) = d_{Te}(x) \cdot d_{Ve}(x)$ is maximized inside the experimental region. The predicted mean values are similar to solution 1. However, it can be observed that the desirability based solution does not regard the variance. It returns a solution where the value for distance is at the border of the experimental region. This is reflected in the predicted variance which is greater than the solution based on the JOP method for both temperature and velocity.

5. Conclusion

In this article we presented the R package **JOP** and demonstrated its usage based on a real data example coming from a thermal spraying process. In its current version, **JOP** can build double generalized linear models for continuous responses and it finds parameter settings for which desired target values of the responses are reached with only small variance.

In future we plan to extend the JOP method in order to deal with correlated responses. This will be also included in the **JOP** package. Furthermore, we want to combine the joint optimization plot with desirabilities.

Acknowledgments

Financial support of the Deutsche Forschungsgemeinschaft (SFB 823, project B1) is gratefully acknowledged.

References

Aitkin M (1987). "Modelling Variance Heterogeneity in Normal Regression Using **GLIM**." *Journal of the Royal Statistical Society C*, **36**(12), 332–339.

Derringer G, Suich R (1980). "Simultaneous Optimization of Several Response Variables." *Journal of Quality Technology*, **12**, 214–219.

Dunn PK, Smyth GK (2012). ***dglm:** Double Generalized Linear Models.* R package version 1.6.2, URL http://CRAN.R-project.org/package=dglm.

Engel J, Huele AF (1996). "A Generalized Linear Modeling Approach to Robust Design." *Technometrics*, **38**, 365–373.

Erdbrügge M, Kuhnt S, Rudak N (2011). "Joint Optimization of Independent Multiple Responses." *Quality and Reliability Engineering International*, **27**(5), 689–703.

Ghalanos A, Theussl S (2012). ***Rsolnp**: General Non-Linear Optimization Using Augmented Lagrange Multiplier Method.* R package version 1.14, URL http://CRAN.R-project.org/package=Rsolnp.

He Z, Wang J, Oh J, Park SH (2010). "Robust Optimization for Multiple Responses using Response Surface Methodology." *Applied Stochastic Models in Business and Industry*, **26**, 157–171.

Hillermeier C (2001). *Nonlinear Multiobjective Optimization.* Birkhäuser Verlag, Basel.

Khuri AI, Mukhopadhyay S (2010). "Response Surface Methodology." *Wiley Interdisciplinary Reviews: Computational Statistics*, **2**(2), 128–149.

Köksoy O (2005). "Dual Response Optimization: The Desirability Approach." *International Journal of Industrial Engineering*, **12**(4), 335–342.

Kuhn M (2012). ***desirability**: Desirabiliy Function Optimization and Ranking.* R package version 1.05, URL http://CRAN.R-project.org/package=desirability.

Kuhnt S, Erdbrügge M (2004). "A Strategy of Robust Parameter Design for Multiple Responses." *Statistical Modelling*, (4), 249–264.

Kuhnt S, Rudak N (2013). ***JOP**: Joint Optimization Plot.* R package version 3.6, URL http://CRAN.R-project.org/package=JOP.

Lenth RV (2009). "Response-Surface Methods in R, Using **rsm**." *Journal of Statistical Software*, **32**(7), 1–17. URL http://www.jstatsoft.org/v32/i07/.

McCullagh P, Nelder JA (1989). *Generalized Linear Models*. 2nd edition. Chapman & Hall, London.

Pignatiello JJ (1993). "Strategies for Robust Multiresponse Quality Engineering." *IIE Transactions*, **25**(3), 5–15.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Roth T (2013). ***qualityTools**: Statistics in Quality Science.* R package version 1.54, URL http://CRAN.R-project.org/package=qualityTools.

Shen LJ, Zhao Y, Yang J (2010). "Mean Square Error Criteria to Multiple Quality Characteristics Robust Design by the Weighted Tchebycheff Method." *Advanced Materials Research*, **118-120**, 881–885.

Smyth GK (1989). "Generalized Linear Models with Varying Dispersion." *Journal of the Royal Statistical Society B*, **51**(1), 47–60.

Smyth GK, Verbyla AP (1999). "Adjusted Likelihood Methods for Modelling Dispersion in Generalized Linear Models." *Environmetrics*, **10**(6), 695–709.

Trautmann H, Steuer D, Mersmann O (2012). ***desire**: Desirability Functions.* R package version 1.06, URL http://CRAN.R-project.org/package=desire.

Vining GG (1998). "A Compromise Approach To Multiresponse Optimization." *Journal of Quality Technology*, **30**, 309–313.

Wu FC (2009). "Robust Design of Nonlinear Multiple Dynamic Quality Characteristics." *Computers and Industrial Engineering*, **56**, 1328–1332.

A. Data sets

Run	X1	X2	Run	X1	X2	Run	X1	X2
1	1.00	1.00	13	−1.00	−1.00	25	0.00	1.41
2	1.00	1.00	14	−1.00	−1.00	26	0.00	1.41
3	1.00	1.00	15	−1.00	−1.00	27	0.00	1.41
4	1.00	1.00	16	−1.00	−1.00	28	0.00	1.41
5	1.00	−1.00	17	−1.41	0.00	29	0.00	0.00
6	1.00	−1.00	18	−1.41	0.00	30	0.00	0.00
7	1.00	−1.00	19	−1.41	0.00	31	0.00	0.00
8	1.00	−1.00	20	−1.41	0.00	32	0.00	0.00
9	−1.00	1.00	21	1.41	0.00	33	0.00	−1.41
10	−1.00	1.00	22	1.41	0.00	34	0.00	−1.41
11	−1.00	1.00	23	1.41	0.00	35	0.00	−1.41
12	−1.00	1.00	24	1.41	0.00	36	0.00	−1.41

Table 6: Exemplary data set `datax`.

Run	Y1	Y2	Run	Y1	Y2	Run	Y1	Y2
1	7.997	0.067	13	33.657	0.063	25	1.341	0.065
2	20.360	0.083	14	37.679	0.064	26	10.775	0.092
3	19.129	0.067	15	35.112	0.047	27	9.073	0.058
4	31.045	0.081	16	43.949	0.043	28	21.835	0.080
5	38.636	0.061	17	7.279	0.054	29	15.446	0.066
6	38.603	0.057	18	21.799	0.074	30	30.339	0.070
7	36.780	0.039	19	19.130	0.050	31	29.767	0.057
8	44.506	0.038	20	36.341	0.063	32	35.113	0.062
9	0.975	0.061	21	23.834	0.066	33	35.027	0.052
10	9.125	0.078	22	31.616	0.080	34	45.159	0.046
11	8.507	0.054	23	35.014	0.065	35	46.927	0.030
12	21.357	0.063	24	36.550	0.055	36	42.352	0.033

Table 7: Exemplary data set `datay`.

Run	L	K	D	FDV	Te	Ve
1	1	−1	1	−1	1575	697
2	1	1	1	1	1648	763
3	−1	−1	1	−1	1588	671
4	−1	−1	−1	1	1629	682
5	0	0	0	0	1646	737
6	0	0	0	0	1637	732
7	−1	1	1	−1	1646	752
8	−1	1	−1	1	1681	757
9	1	1	−1	1	1667	773
10	1	−1	−1	−1	1593	703
11	0	0	0	0	1621	718
12	−1	1	−1	−1	1664	756
13	1	1	−1	−1	1645	771
14	−1	1	1	1	1630	724
15	1	−1	1	1	1572	669
16	−1	−1	1	1	1572	641
17	−1	−1	−1	−1	1607	676
18	1	1	1	−1	1619	754
19	0	0	0	0	1622	716
20	1	−1	−1	1	1601	688
21	0	0	0	0	1619	715
22	0	0	−2	0	1656	738
23	−2	0	0	0	1621	698
24	2	0	0	0	1602	739
25	0	0	0	0	1608	719
26	0	0	0	−2	1610	732
27	0	0	2	0	1567	687
28	0	2	0	0	1656	780
29	0	0	0	2	1613	707
30	0	−2	0	0	1529	628

Table 8: Central composite design together with responses for thermal spraying process.

Affiliation:

Sonja Kuhnt, Nikolaus Rudak
Faculty of Statistics
TU Dortmund University
44221 Dortmund, Germany
E-mail: kuhnt@statistik.tu-dortmund.de, rudak@statistik.tu-dortmund.de

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 54, Issue 9 *Submitted:* 2011-12-07
August 2013 *Accepted:* 2013-03-01