Engage Project Technical Report: Surrogate Models for Maximizing Net Present Value of Renewable Energy Sources

Bastien Talgorn and Michael Kokkolaras
Department of Mechanical Engineering
McGill University

November 9, 2016

Contents

1 Overview of surrogate modeling techniques

From a set of observations $[\mathbf{X}, \mathbf{y}]$, with $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_p\} \subset \mathbb{R}^n$ and $\mathbf{y} = y(\mathbf{X})$ (i.e.: $\mathbf{y}_i = y(\mathbf{x}_i)$) where $y : \mathbb{R}^n \to \mathbb{R}$ is either the objective function f or one of the constraint functions $\{c_j\}_{j \in J}$, it is

possible to build a surrogate model $\hat{y}$ which can be used to predict the value of $y(\mathbf{x})$ for $\mathbf{x} \notin \mathbf{X}$. In this section, we describe three types of surrogate models and how to use them to build an ensemble of surrogates: Polynomial Response Surfaces (PRS) [?, ?, ?], Kernel Smoothing (KS) [?, ?], and Radial Basis Functions (RBF) [?, ?, ?, ?, ?].
To quantify the quality of a surrogate model, it is useful to compute, for each training point $\mathbf{x}_i \in \mathbf{X}$, the value that $\hat{y}(\mathbf{x}_i)$ would have taken if the model $\hat{y}$ had been built without the observation $[\mathbf{x}_i, y(\mathbf{x}_i)]$. Thus, we define $\hat{y}^{(\text{-}i)}$ the surrogate model built by leaving out the observation $[\mathbf{x}_i, y(\mathbf{x}_i)]$ for $\mathbf{x}_i \in \mathbf{X}$. We also define the cross-validation vector $\hat{\mathbf{y}}^{cv}$ such that $\hat{y}^{cv}_i = \hat{y}^{(\text{-}i)}(\mathbf{x}_i)$.

1.1 Linear models

A model $\hat{y}$ is said to be linear if it can be expressed as a linear combination of basis functions [?, ?, ?, ?]:

$$\hat{y}(\mathbf{x}) = \sum_{j=1}^{q} \alpha_j h_j(\mathbf{x}), \tag{1}$$

where $h_j(x) : \mathbb{R}^n \to \mathbb{R}$ is a (possibly non-linear) basis function. The coefficients $\boldsymbol{\alpha} = [\alpha_1, ..., \alpha_q]^\top$ are computed to minimize the regularized quadratic error of the model

$$\sum_{\mathbf{x} \in \mathbf{X}} (y(\mathbf{x}) - \hat{y}(\mathbf{x}))^2 + r_{ridge} \|\boldsymbol{\alpha}\|_2^2,$$

where $r_{ridge} \geq 0$ is a regularization (or ridge [?]) parameter. This parameter ensures that the linear system is invertible, even if there are more basis functions than training points or if the training points are aligned. In the case where the number of training points does not exceed the number of basis functions ($p \leq q$), the model is constructed only if the ridge coefficient is not null. Otherwise, the model will be built only when more training points become available.
For such models, the *design matrix* $\mathbf{H}$ can be built from the training points $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_p\}$:

$$\mathbf{H} = \begin{bmatrix} h_1(\mathbf{x}_1) & ... & h_q(\mathbf{x}_1) \\ \vdots & & \vdots \\ h_1(\mathbf{x}_p) & ... & h_q(\mathbf{x}_p) \end{bmatrix}. \tag{2}$$

Note that the matrix $\mathbf{H}$ is independent from the output $y(\mathbf{X})$. To solve this kind of problem, we consider only the cases where the number of training points is larger than the number of basis functions (i.e. the number of unknown coefficients) or where the ridge coefficient is not null. In these two cases, the use of ordinary least squares (OLS) is required. The coefficients $\boldsymbol{\alpha}$ are then the solution of the invertible problem:

$$\underbrace{(\mathbf{H}^\top \mathbf{H} + r_{ridge} \mathbf{I}_q)}_{\mathbf{A}} \boldsymbol{\alpha} = \mathbf{H}^\top \mathbf{y}. \tag{3}$$

It follows that

$$\boldsymbol{\alpha} = \mathbf{A}^{-1} \mathbf{H}^\top \mathbf{y}. \tag{4}$$

An interesting property of this kind of model is that the cross-validation vector can easily be computed by

$$\hat{\mathbf{y}}^{cv} = \mathbf{y} - diag(\mathbf{P})^{-1} \mathbf{P} \mathbf{y}, \tag{5}$$

where $\mathbf{P}$ is the *projection* matrix [?] such that $\mathbf{P} = \mathbf{I}_p - \mathbf{H}\mathbf{A}^{-1}\mathbf{H}^\top$ and $diag(\mathbf{P})$ is the diagonal matrix such that $diag(\mathbf{P})_{ii} = P_{ii}$. Once the design matrices $\mathbf{H}$ and $\mathbf{A}$ are built and $\mathbf{A}^{-1}$ has been computed for one given function y, it is really inexpensive to compute the coefficients $\boldsymbol{\alpha}$ for other functions $\mathbf{y}$ (for example, each output of the blackbox) and to compute the cross-validation values.

1.2 Polynomial Response Surfaces

Polynomial response surfaces are linear models for which the basis function are polynomials. For a PRS of degree d, the set of basis functions $\{h_j\}_{j=1,\dots,q}$ is a basis of the polynomial vector space of degree d in $\mathbb{R}^n$.

1.3 Radial Basis Functions

RBF models rely on basis functions which follow the form $h_j(\mathbf{x}) = \phi(d(\mathbf{x}_j, \mathbf{x}))$, where d is a distance function (in our case, the Euclidean distance) and $\phi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a kernel function. **[EDIT]** Note that kernel functions are general mathematical concepts that are also used in other modeling techniques like Kernel Smoothing or Kriging. In most studies on RBFs [?, ?, ?, ?, ?], the coefficient of the RBF models are obtained by solving the square linear system

$$\begin{bmatrix} \mathbf{H}^{RBF} & \mathbf{H}^{PRS} \\ (\mathbf{H}^{PRS})^\top & 0 \end{bmatrix} \boldsymbol{\alpha} = \begin{bmatrix} \mathbf{y} \\ 0 \end{bmatrix}, \qquad (6)$$

where $\mathbf{H}^{RBF}$ is a symmetric matrix such that $H_{ij} = \phi(d(\mathbf{x}_i, \mathbf{x}_j))$ and $\mathbf{H}^{PRS}$ is the design matrix of a PRS of degree 1, as described in the previous section.
The main flaw of this method is that the computational cost of building an RBF model can become prohibitive for a large number of training points. **[EDIT]** We propose a novel class of RBF models, denoted RBFI where the "I" stands for "Incomplete basis". This class of models relies on the reduction of the number of basis functions by carefully selecting a subset of the training points $S(\mathbf{X}) = \{\mathbf{x}_i^s\}_{i=1,\dots,q^{RBF}} \subset \mathbf{X}$. This subset contains $q^{RBF} < p$ training points of $\mathbf{X}$ on which will be centered the radial basis functions. The matrix $\mathbf{H}^{RBF}$ is then defined in $\mathbb{R}^{p \times q^{RBF}}$ such that $H_{ij}^{RBF} = \phi(d(\mathbf{x}_i, \mathbf{x}_j^s))\ \forall i = 1, \dots, p\ \forall j = 1, \dots, q^{RBF}$. The design matrix is defined as $\mathbf{H} = [\mathbf{H}^{RBF}\ \mathbf{H}^{PRS}]$, where $\mathbf{H}^{PRS}$ is the PRS design matrix of size $p \times q^{PRS}$. Consequently, the total number of basis function in this model is $q = q^{RBF} + q^{PRS}$. We chose $q^{RBF} = \min\{p/2, 10n\}$. That means that only q^{RBF} radial basis function are selected but they are merged with q^{PRS} PRS basis functions. The PRS used in this work is of degree 1. That means that $q^{PRS} = n + 1$. As the system is overdetermined, we don't need to add orthogonality constraints like in equation (6). The coefficient $\boldsymbol{\alpha}$ are computed with normal equations as described in Section 1.1.
To build the set $S(\mathbf{X})$, we use a greedy algorithm that is computationally efficient and robust in selecting p_S points in the set $\mathbf{X}$. This novel algorithm is described in Algorithm 1. It takes as an input the training set $\mathbf{X}$, the incumbent solution x^*, and the number of selected kernels is set to $\min\{p/2, 100\}$.
The goal of this algorithm is to select a subset S_{out} of p_{out} points from the set $S_{in} \in \mathbb{R}^n$. The set S_{out} must be spread as widely as possible in $\mathbb{R}^n$ and yet we want to favor points of S_{in} that are close to a target point $\mathbf{x}_0$. As these two are conflicting, a trade-off parameter λ is introduced. When many points that are close to $\mathbf{x}_0$ have been selected, it is possible that the pressure to select more points close to $\mathbf{x}_0$ will lead to selecting a point that is already selected (leading to $d(\mathbf{x}_{new}, S_{out}) = 0$). In that case, the trade-off coefficient λ is decreased. An example of training points selection is shown in Figure 1.

Algorithm 1 Greedy selection

```
Input : $S_{in}$, $\mathbf{x}_0$ and $p_{out}$
[1] Initialization
    Randomly draw $\mathbf{x}_{new}$ in $S_{in}\backslash\{\mathbf{x}^*\}$
    $S_{out} \leftarrow \{\mathbf{x}_{new}\} \cup (\mathbf{x}_0 \cap S_{in})$
    $\lambda \leftarrow 3$
    $\lambda_{min} \leftarrow 0.01$
[2] Greedy selection
    while card($S_{out}$) $< p_{out}$ and $\lambda > \lambda_{min}$
        Select $\mathbf{x}_{new} \in \underset{\mathbf{x}\in S_{in}}{\text{argmax}}\ d(\mathbf{x}, S_{out}) - \lambda d(\mathbf{x}, \mathbf{x}_0)$
        if $d(\mathbf{x}_{new}, S_{out}) = 0$
            $\lambda \leftarrow 0.99\lambda$
        then
            $S_{out} \leftarrow S_{out} \cup \{\mathbf{x}_{new}\}$
        end
    end
Return $S_{out}$
```

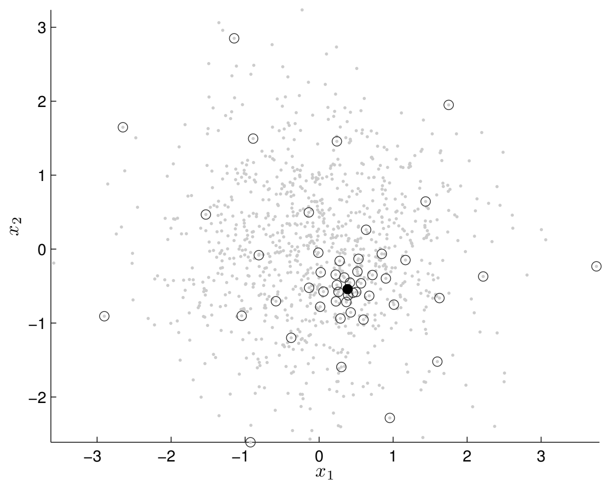

Figure 1: Selection of 100 points out of 1000 points following a Gaussian distribution in $\mathbb{R}^2$. The circles are the selected points.

1.4 Kernel Smoothing

Kernel smoothing models consist of a weighted sum of the training points, where the weight decreases with the distance to the training point:

$$\hat{y}(\mathbf{x}) = \frac{\sum_{i=1}^{n} \phi(d(\mathbf{x}, \mathbf{x}_i))y_i}{\sum_{i=1}^{n} \phi(d(\mathbf{x}, \mathbf{x}_i))}. \tag{7}$$

The advantage of KS is that the computation is immediate. It does not require a linear system inversion. One of the drawbacks is that KS rarely respects the training set, and has a tendency to "undershoot", that is to say that low values will be represented higher than expected and high values will be lower. However, despite their tendency to undershoot, we observe that KS models typically tend to respect the order or sign of the output.

1.5 Ensemble of surrogates

For each (objective or constraint) function y that needs to be computed, we build a set of k_{max} surrogate models $\hat{y}_k$. Then, these models are aggregated into a single model using

$$\hat{y}(\mathbf{x}) = \sum_{k=1}^{k_{max}} w_k \hat{y}_k(\mathbf{x}), \tag{8}$$

where $\mathbf{w} = [w_1, ..., w_{k_{max}}]$ is a weight vector such that $w_k \geq 0$ and $\sum_{k=1}^{k_{max}} w_k = 1$. **[EDIT]** To denote the difference between the ensemble of models (which is itself a model) and the models that comprise this ensemble, we will say, as a convention, that the ensemble of models is made of "simple" models.
Many approaches have been proposed in previous studies to choose the weights $\mathbf{w}$ for a given set of data $[\mathbf{X}, y(\mathbf{X})]$. Most of them [**?**, **?**, **?**] rely on calculating an error metric $\mathcal{E}_k$ for each surrogate $\hat{y}_k$, then on setting $w_k \propto g(\mathcal{E}_k) \geq 0$, where g depends on the method chosen. This notation is equivalent to the two-step definition: $w'_k = g(\mathcal{E}_k)$, then $w_k = \frac{w'_k}{\sum_k w'_k}$.
For example, [**?**] proposes the following weight selection methods

$$\begin{aligned} \text{WTA1}: &\quad w_k \propto \mathcal{E}_{sum} - \mathcal{E}_k \\ \text{WTA2}: &\quad w_k \propto \mathbb{1}_{\mathcal{E}_k = \mathcal{E}_{min}} \\ \text{WTA3}: &\quad w_k \propto (\mathcal{E}_k + \alpha \mathcal{E}_{mean})^{\beta}, \end{aligned}$$

where $\mathcal{E}_{sum}$, $\mathcal{E}_{min}$ and $\mathcal{E}_{mean}$ are respectively the sum, minimum and mean of $\{\mathcal{E}_k\}_{k=1...k_{max}}$. For WTA3, the values $\alpha < 1$ and $\beta < 0$ must be given by the user and [**?**] recommends $\alpha = 0.05$ and $\beta = -1$.
The WTA2 method is tantamount to selecting the best model and, if there are several surrogates with the minimal error, sharing the weights between these models.
Another approach is to optimize the weights to minimize the error metric $\mathcal{E}$ of the aggregated model. In this report, the first method used in choosing the vector $\mathbf{w}$ is to minimize

$$\frac{RMSECV(\mathbf{w})}{1 - \mathbb{V}\text{ar}(\mathbf{w})},$$

where RMSECV (Root mean Square Error with Cross-Validation, also called PRESS (Predicted REsidual Sum of Squares)) [**?**, **?**] is the quadratic cross-validation error

$$RMSECV(\mathbf{w}) = \sqrt{\frac{1}{p} \sum_{i=1}^{p} \left(y(\mathbf{x}_i) - \sum_{k=1}^{k_{max}} w_k \hat{y}_k^{(\text{-}i)}(\mathbf{x}_i) \right)^2}, \tag{9}$$

where $\hat{y}_k^{(\text{-}i)}(\mathbf{x}_i)$ is the cross-validation value of the simple model $\hat{y}_k$ in $\mathbf{x}_i$ and

$$\mathbb{V}\text{ar}(\mathbf{w}) = \frac{k_{max}\left(\sum_{k=1}^{k_{max}} w_k^2\right) - 1}{k_{max} - 1} \tag{10}$$

is used to favor the use of several models. The second method consists of minimizing

$$\frac{1}{\mathcal{L}(\mathbf{w})(1 - \mathbb{V}\text{ar}(\mathbf{w}))},$$

where $\mathcal{L}$ is the likelihood of the model, defined as

$$\begin{aligned}\mathcal{L} &= \prod_{\mathbf{x}\in\mathbf{X}} \mathbb{P}[y(\mathbf{x})|\hat{y}(\mathbf{x}),\hat{\sigma}^2(\mathbf{x})] \\ &= \prod_{\mathbf{x}\in\mathbf{X}} \frac{1}{\sqrt{2\pi}\hat{\sigma}(\mathbf{x})} \exp\left(\frac{(y(\mathbf{x})-\hat{y}(\mathbf{x}))^2}{\hat{\sigma}^2(\mathbf{x})}\right). \end{aligned} \tag{11}$$

Note that is is generally more practical and numerically-stable to compute

$$\log\mathcal{L} \propto \sum_{\mathbf{x}\in\mathbf{X}} -\log(\hat{\sigma}(\mathbf{x})) + \frac{(y(\mathbf{x})-\hat{y}(\mathbf{x}))^2}{\hat{\sigma}^2(\mathbf{x})}$$

The models used in the ensemble of surrogates are listed in Table 1.

Table 1: List of simple surrogate models

#	Model type	Param. 1	Param. 2
1	PRS	Degree = 1	r_{ridge} = 0
2		1	10^{-3}
3		2	0
4		2	10^{-3}
5		3	0
6		6	10^{-3}
7	KS	r_ϕ = 0.1	Gaussian Kernel
8		0.3	
9		1.0	
10		3.0	
11		10	
12	RBFI	r_ϕ = 0.3	Gaussian Kernel
13		1.0	
14		3.0	
15		10	
16		Degree = 1	PolyHarmonic Kernel
17		2	

1.6 Closest Neighboor

1.7 Locally Weighted Regression

2 Model definition

The following section explains how to define the surrogate model used in the search. The definition of the model must be provided in the Nomad parameter file after the option `SGTELIB_MODEL_DEFINITION` . The model definition is made of several field names followed by their value. For example:

$$\texttt{SGTELIB\_MODEL\_DEFINITION} \quad \underbrace{\texttt{TYPE}}_{\text{Field 1}} \quad \underbrace{\texttt{PRS}}_{\substack{\text{Value of} \\ \text{field 1}}} \quad \underbrace{\texttt{DEGREE}}_{\text{Field 2}} \quad \underbrace{\texttt{2}}_{\substack{\text{Value of} \\ \text{field 2}}}$$

defines a model of type PRS and of degree 2. The field names can be one of the following:

Field name	Use
`TYPE`	Defines which type of model is used
`DEGREE`	Defines the degree of a polynomial response surface
`RIDGE`	Defines the regularization value of polynomial response surface
`KERNEL`	Defines the type of kernel function
`KERNEL_TYPE`	Idem as `KERNEL`
`KERNEL_COEF`	Define the shape coefficient of a kernel function
`SHAPE_COEF`	Idem as `SHAPE_COEF`
`DISTANCE`	Defines which distance metric is used in kernel methods
`WEIGHT`	For Ensembles of Models, define the method used to compute the weights
`METRIC`	For Ensembles of Models, define the metric considered to compute the weights
`PRESET`	For Ensembles of Models, define which simple models are used

All these keywords are described hereafter.

2.1 TYPE

The keyword "`TYPE`" defines which type of model is used.
Possible values:

Value	Comment
`PRS`	Polynomial Response Surface
`KS`	Kernel Smoothing
`PRS_EDGE`	PRS EDGE model
`PRS_CAT`	PRS CAT model
`RBF`	Radial Basis Function Model
`RBFI`	RBF model with incomplete basis
`LWR`	Locally Weighted Regression
`ENSEMBLE`	Ensemble of surrogates
`DYNATREE`	dynaTree model **(not supported yet)**
`TGP`	TGP model **(not supported yet)**
`KRIGING`	Kriging model **(not supported yet)**

Default value: This parameter does not have a default value. It must always be defined.

Example:
"`TYPE PRS`" defines a PRS model.
"`TYPE ENSEMBLE`" defines an ensemble of models.

2.2 DEGREE

The keyword "`DEGREE`" defines the degree of a polynomial response surface.
Allowed for models of type: PRS, PRS_EDGE, PRS_CAT, LWR, RBFI.
Possible values: The value must be an integer ≥ 1.
Default values: DEGREE = 1 for models of type RBFI. DEGREE = 2 for models of type PRS, PRS_EDGE, PRS_CAT and LWR.
Example:
"`TYPE PRS DEGREE 3`" defines a PRS model of degree 3.
"`TYPE PRS_EDGE DEGREE 2`" defines a PRS_EDGE model of degree 2.

2.3 RIDGE

The keyword "`RIDGE`" defines the regularization parameter of the model.
Allowed for models of type: PRS, PRS_EDGE, PRS_CAT, LWR, RBFI.
Possible values: Real value ≥ 0. Recommended values are 0 and 0.001.
Default values: Default value is 0.01.
Example:
"`TYPE PRS DEGREE 3 RIDGE 0`" defines a PRS model of degree 3 with no regularization.

2.4 KERNEL_TYPE

The keyword "`KERNEL_TYPE`" defines the type of kernel used in the model. The keyword"`KERNEL`" is equivalent.
Allowed for models of type: RBF, RBFI, KS.
Possible values:

Value	Comment
`D1`	Gaussian kernel $\phi(d) = \exp\left(\frac{r_\phi^2 d^2}{d_{mean}^2}\right)$
`D2`	Inverse Quadratic Kernel, $\phi(d) = \left(1 + \frac{r_\phi^2 d^2}{d_{mean}^2}\right)^{-1}$
`D3`	Inverse Multiquadratic Kernel, $\phi(d) = \left(1 + \frac{r_\phi^2 d^2}{d_{mean}^2}\right)^{-1/2}$
`I0`	Multiquadratic Kernel, $\phi(d) = \sqrt{1 + \frac{r_\phi^2 d^2}{d_{mean}^2}}$
`I1`	Polyharmonic splines, $k = 1$, $\phi(d) = d$
`I2`	Polyharmonic splines, $k = 2$, $\phi(d) = d^2 \log(d)$
`I3`	Polyharmonic splines, $k = 3$, $\phi(d) = d^3$
`I4`	Polyharmonic splines, $k = 4$, $\phi(d) = d^4 \log(d)$

Default values: Default value is "`D1`".
Example:
"`TYPE KS KERNEL_TYPE D1`" defines a KS model with Inverse Quadratic Kernel.

2.5 KERNEL_COEF

The keyword "`KERNEL_COEF`" defines the shape coefficient r_ϕ of the kernel function. Note that this keyword has no impact for KERNEL_TYPES I1, I2, I3 and I4 because this kernels do not depend on r_ϕ.
Alternative keyword name: `SHAPE_COEF`
Allowed for models of type: RBF, RBFI, KS.
Possible values: Real value ≥ 0. Recommended range is $[0.1, 10]$. Small values lead to smoother models.
Default values: Default value is 5.
Example:
"`TYPE RBF KERNEL_COEF 10`" defines a RBF model with $r_\phi = 10$.

2.6 DISTANCE_TYPE

The keyword "`DISTANCE_TYPE`" defines the distance function used in the model. The keyword "`SHAPE_COEF`" is equivalent.
Allowed for models of type: RBF, RBFI, KS.
Possible values:

Value	Comment
`NORM1`	Euclidian distance
`NORM2`	Distance based on norm 1
`NORMINF`	Distance based on norm ∞
`NORM2_IS0`	Tailored distance for discontinuity in 0.
`NORM2_CAT`	Tailored distance for categorical models.

Default values: Default value is "`NORM2`".
Example:
"`TYPE KS DISTANCE NORM2_IS0`" defines a KS model tailored for VAN optimization.

2.7 WEIGHT

The keyword "`WEIGHT`" defines the method used to compute the weights $\mathbf{w}$ of the ensemble of models. The keyword "`WEIGHT_TYPE`" is equivalent.
Allowed for models of type: ENSEMBLE.
Possible values:

Value	Comment
`WTA1`	$w_k \propto \mathcal{E}_{sum} - \mathcal{E}_k$
`WTA3`	$w_k \propto (\mathcal{E}_k + \alpha \mathcal{E}_{mean})^\beta$
`SELECT`	$w_k \propto \mathbb{1}_{\mathcal{E}_k = \mathcal{E}_{min}}$
`OPTIM`	$\mathbf{w}$ minimizes $\mathcal{E}(\mathbf{w})/(1 + \mathbb{V}\text{ar}(\mathbf{w})$
`NORM2_IS0`	Tailored distance for discontinuity in 0.

Default values: Default value is WTA1.
Example:
"`TYPE ENSEMBLE WEIGHT SELECT METRIC RMSECV`" defines an ensemble of models which selects the model that has the best RMSECV.

"`TYPE ENSEMBLE WEIGHT OPTIM METRIC RMSECV`" defines an ensemble of models where the weights $\mathbf{w}$ are computed by minimizing RMSECV/$(1 - \mathbb{V}\text{ar}(\mathbf{w}))$.

2.8 METRIC

The keyword "`METRIC`" defines the metric used to compute the weights $\mathbf{w}$ of the ensemble of models. The keyword"`METRIC_TYPE`" is equivalent.
Allowed for models of type: ENSEMBLE.
Possible values:

Value	Comment
`EMAX`	Error Max.
`EMAXCV`	Error Max with Cross-Validation.
`RMSE`	Root Mean Square Error.
`RMSECV`	RMSE with Cross-Validation.
`OE`	Order Error.
`OECV`	Order Error with Cross-Validation.
`LINV`	Inverte of the Likelihood.

Default values: Default value is "`RMSE`".
Example:
"`TYPE ENSEMBLE WEIGHT SELECT METRIC RMSECV`" defines an ensemble of models which selects the model that has the best RMSECV.

2.9 PRESET

The keyword "`PRESET`" defines the set of simple models used to build the ensemble of models.
Allowed for models of type: ENSEMBLE.
Possible values:

Value	Comment
`DEFAULT`	Set of 17 models (PRS, KS and RBFI).
`KS`	Set of 7 KS models with Gaussian kernels.
`PRS`	Set of 6 PRS models (degree 1 to 6).
`SMALL`	Set of 3 models (PRS, KS and RBFI) with default parameters.
`IS0`	Set of 30 models that handles the discontinuity in 0.
`CAT`	Set of 30 models that handles categorical data.

The preset "`IS0`" is recommended for VAN problems. The preset "`DEFAULT`" is recommended for other problems.
Default values: Default value is "`DEFAULT`".
Example:
"`TYPE ENSEMBLE PRESET DEFAULT`" defines the default ensemble of models.
"`TYPE ENSEMBLE PRESET IS0`" defines an ensemble of models tailored for VAN optimization.
"`TYPE ENSEMBLE PRESET CAT`" defines an ensemble of models tailored the 2D problem.

3 Interface with matlab

4 References