Electronic Supplementary Information (ESI)

[A] Brief Review of Radial Basis Function Neural Networks.

Suppose that N observations y^i have been made at locations $\mathbf{x}^i = (x^i_1, x^i_2, \ldots, x^i_d)$ i=1,2,...,N. The activation function, denoted φ and acting on the hidden layer in a RBFNN, is a positive-definite function of the distance r to a "RBF centre" $\mathbf{c}^i$ (hence the name "radial basis function")

$$\varphi\left(\mathbf{x},\mathbf{c}^i\right)=\varphi\left(\left\|\mathbf{x}-\mathbf{c}^i\right\|\right)=\varphi(r),\ r>0 \qquad \text{(A1)}$$

where $\mathbf{x}$ is an arbitrary input vector (of feature values), and the centres $\mathbf{c}^i$ are the input vectors present in the training set, i.e. $\mathbf{c}^i = \mathbf{x}^i$, i=1,2,…, N. Many different types of radial basis function exist, for example:

$\varphi(r)=r$ (linear)

$\varphi(r)=r^3$ (cubic)

$\varphi(r)=\sqrt{r^2+\gamma^2},\ (\gamma>0)$ (multiquadric) (A2)

$\varphi(r)=r^2\log r$ (thin plate spline)

along with many others. In this paper thin plate splines are used. With the activation functions in the hidden layer defined, the prediction made by a RBFNN, $\hat{y}(\mathbf{x})$ is written as

$$\hat{y}(\mathbf{x})=\sum_{i=1}^{N}\varphi\left(\mathbf{x},\mathbf{x}^i\right)\beta_i \qquad \text{(A3)}$$

where $\boldsymbol{\beta}$ = (β_1, β_2 ,…, β_N) is the vector of weights defining the linear mapping from the hidden layer to the output. By requiring the prediction to be equal to the true values at the training points,

$$\sum_{i=1}^{N}\varphi\left(\mathbf{x}^j,\mathbf{x}^i\right)\beta_i=y^j,\quad j=1,2,...,N \qquad \text{(A4)}$$

we achieve exact interpolation of the training points. This leads to a system of linear equations

$$\mathbf{\Phi\beta} = \mathbf{y} \qquad \text{(A5)}$$

where **Φ** is the *N* x *N* matrix whose *i*-*j*th element is φ(**x**^{i}, **x**^{j}), and **y** is the *N* x 1 vector whose *i*th entry is *y*^{i}. Simple matrix inversion,

$$\boldsymbol{\beta} = \mathbf{\Phi}^{-1}\mathbf{y} \qquad \text{(A6)}$$

then gives the weight vector **β.** The non-singularity of **Φ** is guaranteed by Micchelli's theorem[1].

[B] Brief Review of Kriging.

Experts in computational chemistry, not familiar with Kriging, will benefit from the following compact, yet detailed and accessible account, which is based on Jones' exposition in reference[2]. As in Section 2.5, suppose that *N* observations *y*^{i} (of a particular multipole moment) have been made at locations **x**^{i} = [x$^i_1$, x$^i_2$, ..., x^i_d]^T where T denotes the transpose, *i*=1,2,...,*N* and *d* is the dimensionality of the feature space[a]. In our case, **x** is the vector of features describing a particular water cluster configuration. Then these observations are modelled as having been generated from the following model:

$$y(\mathbf{x}^i) = \sum_h \beta_h f_h(\mathbf{x}^i) + \varepsilon^i \qquad i=1,2,...,N \qquad \text{(B1)}$$

Each $f_h(\mathbf{x}^i)$ is a function of the feature space of the problem, where $f(\mathbf{x}^i)$ is a polynomial term of the form $x_1^{g1}x_2^{g2} \ldots x_d^{gd}$ while $g_1+g_2+\ldots+g_d \le g_{max}$ and g_{max} is the order of the polynomial. In this work, the feature space consists of polar and Euler coordinates defining the configuration of water molecules, as detailed in Section 3. The summed term on the right hand side may be seen as a global trend (over feature space) for the observable *y* (in our case a multipole moment). The ε terms may be viewed as normally distributed random variables (with mean zero). They are "error terms" compensating for the inadequacy of the global term in modelling the observed values exactly. Therefore the ε terms are really collections of terms in **x**, and may be written ε(**x**). However, as shall become clear below, this is the focus of the Kriging method.

Now consider two distinct observations y^i and y^j, made at **x**^{i} and **x**^{j}, respectively. If **x**^{i} and **x**^{j} are close together (in feature space), then the errors ε(**x**^{i}) and ε(**x**^{j}) should be close

[a] The vector **x**^{i} is introduced here as a column vector although this point is not essential in the account below.

together – that is to say they are correlated, and this correlation may be parameterised in several different ways. One common way for parameterizing the correlation between the error terms is through the exponential power correlation function:

$$Cor[\varepsilon(\mathbf{x}^i),\varepsilon(\mathbf{x}^j)] = \exp\left[-\sum_{h=1}^{d}\theta_h \mid x_h^i - x_h^j \mid^{p_h}\right] \qquad \text{(B2)}$$

where $\theta_h > 0$ and $0 < p_h \leq 2$. This correlation function is a N x N matrix whose i-jth entry is Cor[$\varepsilon(\mathbf{x}^i)$,$\varepsilon(\mathbf{x}^j)$]. It is convenient to denote this matrix by **R**. This correlation function has the intuitive property that if the distance between $\mathbf{x}^i$ and $\mathbf{x}^j$ is small then the correlation is close to one. On the other hand, if the distance between $\mathbf{x}^i$ and $\mathbf{x}^j$ is large, the correlation will approach zero. In other words, similar water configurations have similar multipole moment values.

Modelling the correlation in this way is so effective that the global terms in eq.(B1) may be replaced by a single constant term. Thus the observations are modelled as having been generated from the following model:

$$y(\mathbf{x}^i) = \mu + \varepsilon(\mathbf{x}^i) \qquad i=1,2,...,N \qquad \text{(B3)}$$

where μ models the global trend of the observable y, and $\varepsilon(\mathbf{x}^i)$ is a Gaussian distribution with mean 0 and some variance σ^2, the correlation between observations at $\mathbf{x}^i$ and $\mathbf{x}^j$ being given by Eq.(B2). So there are $2d+2$ parameters to be determined: μ, σ^2, $\boldsymbol{\theta}=(\theta_1,\theta_2,...,\theta_d)$ and $\mathbf{p}=(p_1,p_2,...,p_d)$ where d is the dimension of the feature space (the number of descriptors used to describe a water cluster configuration). These parameters are determined by maximizing the likelihood of the observations in the training set. Denoting the vector of observations by $\mathbf{y}=[y^1, y^2,..., y^N]^T$ and using the correlation matrix **R**, the likelihood of the training data is given by the N-variate Gaussian distribution,

$$L(\boldsymbol{\theta},\mathbf{p},\mu,\sigma \mid y^i;\, i=1,2,...,N) = \frac{1}{(2\pi)^{N/2}(\sigma^2)^{N/2}|\mathbf{R}|^{1/2}}\exp\left[-\frac{(\mathbf{y}-\mathbf{1}\mu)^T\mathbf{R}^{-1}(\mathbf{y}-\mathbf{1}\mu)}{2\sigma^2}\right] \qquad \text{(B4)}$$

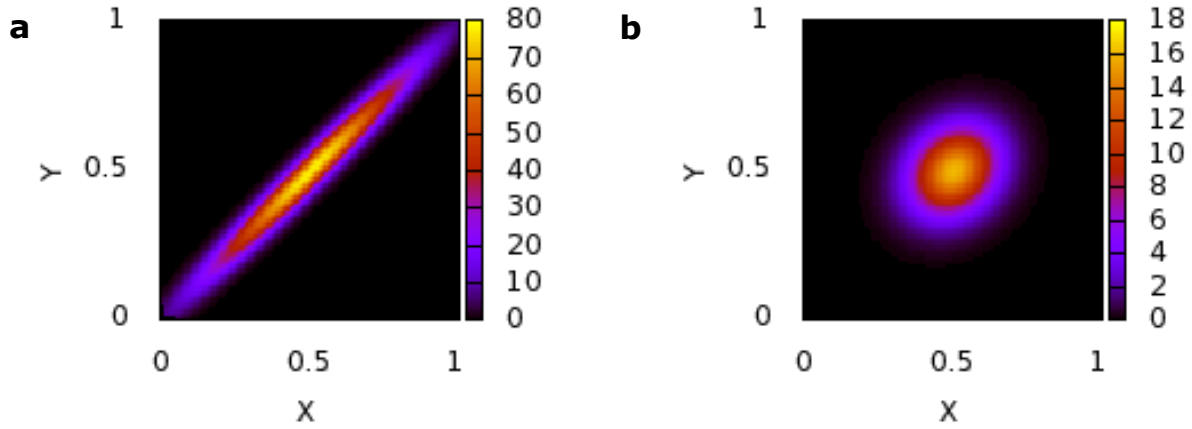
which is the N-dimensional generalisation of the univariate Gaussian distribution. The symbol **1** represents $[1,1,\ldots,1]^T$. Since all computational expense would be in inverting **R**, savings can be made by a LUD decomposition of **R** and evaluating $\mathbf{R}^{-1}(\mathbf{y}-\mathbf{1}\mu)$ in our implementation.

To illustrate the principle, consider the unrealistic case of having just two observations y^1 and y^2 (N=2). Then the likelihood function is given by the bivariate Gaussian distribution, which is shown in Figure B1. The set of two observations is represented by a point in the XY plane, where X and Y are two *random variables.* Observation y^1 is modelled to be an occurrence of random variable X and y^2 of random variable Y. In our case, an observation is a multipole moment value. The dimensionality d is not determined in this example but of

course appears in eq.(B2) and hence is involved in determining Cor(X,Y), which features in this Figure. It is the purpose of maximum-likelihood estimation to vary the parameters of the bivariate Gaussian, such that this point lies in the region of highest likelihood, since the observations are fixed and the parameters are being varied. For example, varying the parameters $\boldsymbol{\theta}$ and **p** influences the correlation between the random variables *X* and *Y* representing our two observations, as shown in Figure B1. Certain values of $\boldsymbol{\theta}$ and **p** lead to a high correlation between *X* and *Y* (Fig. B1(a)), whereas other values lead to a low correlation (Fig. B1(b)). Varying the correlation influences the shape of the likelihood function. Given two observations (i.e. a point in the *XY* plane), $\boldsymbol{\theta}$ and **p** (along with μ and σ^2) are varied to give a likelihood function such that the value of the likelihood at the point representing the two observations is maximised.

Figure B1

Bivariate Gaussian distributions for two random variables *X* and *Y* that are (a) strongly correlated (Cor*(X,Y)* = 0.98) and (b) weakly correlated (Cor*(X,Y)* = 0.2). The colour legend on the right indicates the value of the likelihood function. Note that μ is arbitrarily set to 0.5 in both cases.



Returning to the general case of *N* observations, it is the log-likelihood, readily obtained from eq.(B4),

$$-\frac{N}{2}\log(\sigma^2)-\frac{1}{2}\log(|\mathbf{R}|)-\frac{(\mathbf{y}-\mathbf{1}\mu)^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y}-\mathbf{1}\mu)}{2\sigma^2} \qquad \text{(B5)}$$

which is maximised in practice (note that constant terms have been dropped, as they do not affect the outcome of optimisation). Furthermore, by setting the derivatives of the log-likelihood with respect to σ^2 and μ equal to zero and solving this new equation, the optimal values of σ^2 and μ can be written in terms of $\boldsymbol{\theta}$ and $\mathbf{p}$. Hence, L only needs to be optimised with respect to $\boldsymbol{\theta}$ and $\mathbf{p}$[b]**.** The optimisation of eq.(B5) may be carried out by any means, but popular choices include the Nelder-Mead simplex algorithm[3], genetic algorithms[4], and branch-and-bound algorithms[5]. As the evaluation of eq.(B5) requires the inversion of a N x N matrix, efficiency is of great importance if the number of observations N is large (in this work, N is set to maximum 1000). In this paper, we use Nelder-Mead algorithm, with up to 8 restarts. As Nelder-Mead is a downhill search method, restarting the algorithm after it seems to have converged reduces the risk of having converged to a local optimum of the likelihood function.

Having determined the parameters of the N-variate Gaussian distribution that maximise the likelihood of the observed training data, the question arises of how they may be used to make a prediction $\hat{y}(\mathbf{x}^*)$ of y at an *unevaluated* point $\mathbf{x}^*$ in feature space. Consider adding $(\mathbf{x}^*, \hat{y}(\mathbf{x}^*))$ to the training set, and denote by L^* the likelihood function augmented with this extra observation. Now instead of considering L^* as a function of $\boldsymbol{\theta}$, $\mathbf{p}$, μ and σ with the training data fixed, consider L^* with everything fixed (the training data, the position of $\mathbf{x}^*$, and $\boldsymbol{\theta}$, $\mathbf{p}$, μ, σ at their previous optimal values, $\boldsymbol{\theta}^*$, $\mathbf{p}^*$, μ^*,σ^*) except the value of $\hat{y}(\mathbf{x}^*)$. Then L^* is a function of $\hat{y}(\mathbf{x}^*)$, and a good value of $\hat{y}(\mathbf{x}^*)$ to predict is the value that maximises L^*: this is the value that is most consistent with the pattern of variation observed in the training set.

To aid with notation, denote by $\mathbf{r}$ the vector of correlations of the Gaussian Process at $\mathbf{x}^*$ with the Gaussian Process at the points in the training set $\mathbf{r}=($ $\mathrm{Cor}(\varepsilon(\mathbf{x}^*),\varepsilon(\mathbf{x}_1))$, $\mathrm{Cor}(\varepsilon(\mathbf{x}^*),\varepsilon(\mathbf{x}_2))$,…, $\mathrm{Cor}(\varepsilon(\mathbf{x}^*),\varepsilon(\mathbf{x}_N))$ $)^T$. Then the (N+1)x(N+1) correlation matrix $\tilde{\mathbf{R}}$ for the augmented data set is:

$$\tilde{\mathbf{R}} = \begin{pmatrix} \mathbf{R} & \mathbf{r} \\ \mathbf{r}^T & 1 \end{pmatrix} \qquad \text{(B6)}$$

From Eq.(B5) it can be seen that the only part of the augmented log-likelihood function that depends upon $\hat{y}(\mathbf{x}^*)$ is

$$-\frac{\left(\tilde{\mathbf{y}} - \mathbf{1}\mu^*\right)^{\mathrm{T}} \tilde{\mathbf{R}}^{-1} \left(\tilde{\mathbf{y}} - \mathbf{1}\mu^*\right)}{2\sigma^{*2}} \qquad \text{(B7)}$$

where $\tilde{\mathbf{y}} = (\mathbf{y}^{\mathrm{T}} \ \hat{y}(\mathbf{x}^*))^{\mathrm{T}}$. Substituting in the expressions for $\tilde{\mathbf{y}}$ and $\tilde{\mathbf{R}}$, Eq.(B7) becomes:

[b] Note that in the main text $\boldsymbol{\Theta}$ (capital theta) was used to refer to *all k* Kriging parameters. So, $\boldsymbol{\Theta}=(\boldsymbol{\theta},\mathbf{p})$ and $k=2d$.

$$\frac{-\begin{pmatrix} \mathbf{y}-\mathbf{1}\mu^* \\ \hat{y}(\mathbf{x}^*)-\mu^* \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \mathbf{R} & \mathbf{r} \\ \mathbf{r}^{\mathrm{T}} & 1 \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{y}-\mathbf{1}\mu^* \\ \hat{y}(\mathbf{x}^*)-\mu^* \end{pmatrix}}{2\sigma^{*2}} \qquad \text{(B8)}$$

To determine the value of $\hat{y}(\mathbf{x}^*)$, the derivative of eq.(B8) with respect to $\hat{y}(\mathbf{x}^*)$ needs to be set equal to zero. Using the following identity for the inverse of a partitioned matrix[6]:

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{M} & -\mathbf{MBD}^{-1} \\ -\mathbf{D}^{-1}\mathbf{CM} & \mathbf{D}^{-1}+\mathbf{D}^{-1}\mathbf{CMBD}^{-1} \end{pmatrix} \qquad \text{(B9)}$$

where $\mathbf{M}=(\mathbf{A}-\mathbf{BD}^{-1}\mathbf{C})^{-1}$, the expression for $\widetilde{\mathbf{R}}$, the expression for L^* becomes:

$$\left[\frac{-1}{2\sigma^{*2}\left(1-\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{r}\right)}\right]\left(\hat{y}(\mathbf{x}^*)-\mu^*\right)^2+\left[\frac{\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y}-\mathbf{1}\mu^*)}{\sigma^{*2}\,(1-\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{r})}\right]\left(\hat{y}(\mathbf{x}^*)-\mu^*\right)+\text{terms without } \hat{y}(\mathbf{x}^*) \qquad \text{(B10)}$$

Setting to zero the derivative with respect to $\hat{y}(\mathbf{x}^*)$ of eq.(B10) gives:

$$\left[\frac{-1}{\sigma^{*2}\left(1-\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{r}\right)}\right]\left(\hat{y}(\mathbf{x}^*)-\mu^*\right)+\left[\frac{\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y}-\mathbf{1}\mu^*)}{\sigma^{*2}\,(1-\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{r})}\right]=0 \qquad \text{(B11)}$$

Upon rearranging, this gives an expression for the value of $\hat{y}(\mathbf{x}^*)$,

$$\hat{y}(\mathbf{x}^*)=\mu^*+\mathbf{r}^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{y}-\mathbf{1}\mu^*) \qquad \text{(B12)}$$

This is the master equation, i.e. the Kriging prediction for the value of the observable at $\mathbf{x}^*$. This may be rewritten in the (simpler to program) format:

$$\hat{y}(\mathbf{x}^*)=\mu^*+\sum_{i=i}^{N} a_i\varphi\left(\mathbf{x}^*-\mathbf{x}^i\right) \qquad \text{(B13)}$$

where a_i is the i-th element of $\mathbf{R}^{-1}(\mathbf{y}-\mathbf{1}\mu^*)$, and $\varphi\left(\mathbf{x}^*-\mathbf{x}^i\right)=\exp\left[-\sum_{h=1}^{d}\theta_h\,|\,x_h^*-x_h^i\,|^{p_h}\right]$ is the i-th element of $\mathbf{r}$ (defined just before eq. B6).

References

1 C. A. Micchelli, *Constructive Approximation* 1986, **2**, 11-22.

2 D. R. Jones, *J.Global Optim.*, 2001, **21**, 345-383.

3 J. A. Nelder and R. Mead, *The Computer Journal*, 1965, **7**, 308-313.

4 D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Kluwer Academic Publishers, Boston, MA, 1989.

5 E. L. Lawler and D. E. Wood, *Operations Research*, 1966, **14**, 699-719.

6 C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.