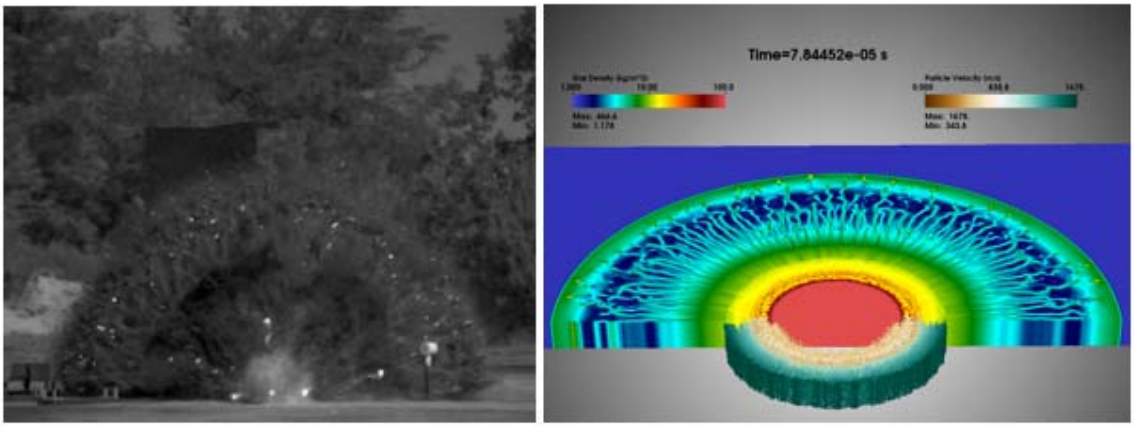
1. Introduction to Surrogate models

1.1. What is surrogate modeling?

In engineering design, a quantity of interest (QoI) is a function of many input parameters or design variables. For example, the maximum stress in a cantilevered beam is a function of the beam length, cross-sectional dimensions, and applied loads. Such a functional relationship is often governed by complex physics, and it may not exist in a simple analytical form. Many engineering mechanics problems are governed by integro-differential equations, which often require computationally intensive numerical methods to find the relationship. Finite element methods, computational fluid dynamics, and rigid-body dynamics comprise a short list of examples of numerical methods used to establish the relationship between a QoI and input parameters or design variables. Even if the modern technology has enhanced computational power significantly, engineers still anticipate a significant amount of computational time to calculate the QoI.

Figure 1-1(a) shows an example of dispersed multiphase flow, where numerous particles interact with shock waves generated from explosives [1]. The physics of shock interaction with a cloud of particles has many interesting applications, such as explosive volcanic eruptions, dust explosions in coal mines, and supernovae [2]. Simulation of shock interaction with distributed particles as shown in Figure 1-1(b) creates many computational challenges. Problems involving distributed particles present a wide range of length scales, from hundreds of nanometers (shock thickness) to meters (size of the domain) and an associated wide range of time scales from microseconds to minutes. Given this wide range of length and time scales, fully resolved particle simulations of dispersed multiphase flows involving a large number of particles are practically impossible, where the complete details of both the flow around the particles and the particle motion are captured. Instead, Euler-Lagrange formations are often adopted with point-particle force and heat transfer models to couple the particle momentum and energy with those of the surrounding fluid. While the governing Navier-Stokes equations are solved in the region outside the particles with a sufficiently high resolution in the Eulerian frame of reference, the motion of each particle is tracked by solving its equations of motion in the Lagrangian frame of reference. Even such an approximate model pushes the limit of current computational resources. The simulation in Figure 1-1(b), for example, took 15,000 seconds with 2,400 clusters in a high-performance computing facility at U.S. National Laboratories.



(a) (b)

Figure 1-1: Experiment and simulation of dispersed multiphase flow: Shock wave interaction with a particle bed. (a) Experiment at Eglin Air Force Base, (b) Simulation at the University of Florida

In many engineering processes, multiple simulations are frequently required by changing input parameters or design variables. To find the optimal airfoil shape, for example, an engineer simulates the airflow around a wing for different design variables, such as length, curvature, aspect ratio, chord length, and even material. During design optimization, design variables are updated at each iteration to improve the objective function while satisfying constraints. Commonly, the design variables are updated more than 100 times before reaching an optimum design. Since the objective function and constraints need to be recalculated with each updated design, numerous simulations are required during the design optimization process. Other examples are design space exploration, sensitivity analysis, what-if analysis, reliability analysis, and uncertainty quantification. All of these examples require numerous simulations by changing input parameters or design variables. As shown in the previous example, if a single simulation is already computationally expensive, it becomes easily impractical to run numerous simulations.

Even if the process of calculating QoI is governed by complex physics and expensive numerical simulations, the relationship between the QoI and input parameters can be simplified. In the case of a cantilevered beam shown in Figure 1-2, for example, the maximum stress can be written as

$$\sigma_{max} = \frac{6FL}{bh^2} \tag{1.1}$$

where F is the applied force at the tip, L the length of the beam, b the width, and h the height of the cross-section. Even if in this simple example the QoI (maximum stress) is expressed in an analytical form, we may consider that it is from complex physics and expensive numerical simulation. However, the relationship between maximum stress and applied force is linear, $\sigma_{max} = \alpha F$, and it is inversely proportional to the height of the beam, $\sigma_{max} = \beta h^{-2}$. Therefore, without considering complex physics, it is possible to express the relationship using simple functions, such as $\sigma_{max} = f(F, L, b, h)$. However, the issues of determining the functional form and calculating the coefficients remain.

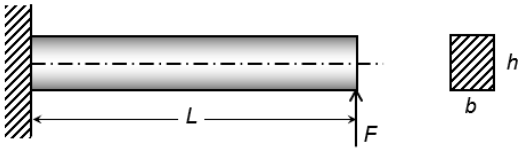


Figure 1-2: Bending of a cantilevered beam

Surrogate modeling is a powerful engineering tool when a QoI cannot be easily calculated (or measured), so an approximation of the QoI is used instead. It mimics (i.e., approximates) the behavior of the simulation model as closely as possible while being computationally cheap in evaluating the QoI. It is a so-called data-driven approach where it does not concern physics or underlying theory. Instead, it only concerns input-output behavior, and the relationship is represented using mathematical functions. This approach is also known as a response surface model [4], meta-model [5], reduced model [6], model emulator [7], proxy model [8], lower-fidelity model [9], behavioral modeling, or black-box modeling in different communities. When only a single input parameter is involved, the modeling process is known as curve-fitting.

1.2. Surrogate models

The idea of surrogate modeling is to represent the relationship between the QoI and input parameters using known functional forms with unknown function coefficients. Different surrogate models can be defined based on how to select the functional forms and how to determine unknown coefficients. However, all of them share some common characteristics. As discussed more detail later, the most important characteristic relates to samples. All surrogate models utilize samples to determine the unknown coefficients. Samples in a surrogate model mean data between input parameters and output QoIs. The dimension of a surrogate model refers to the number of input variables. Figure 1-3(a) shows an example of samples in a two-dimensional surrogate model. Each sample consists of the paired values of input parameters and output QoI. Let input parameters be x_1 and x_2 and output QoI be y. Then, a sample can be represented by $S = (x_1, x_2, y)$. In general, the k-th sample of N-dimensional surrogate can be written as $S^{(k)} = (\mathbf{x}^{(k)}, y^{(k)})$, where $\mathbf{x} = \{x_1, x_2, \dots, x_N\}^{\mathrm{T}}$ is the vector of input sample locations. In this text, a sample refers to $S^{(k)}$, while a sample location refers to $\mathbf{x}^{(k)}$. The example in Figure 1-3(a) has 21 samples. For a given input sample location, $\mathbf{x}^{(k)}$, it is necessary to evaluate the output QoI sample $y^{(k)}$, which requires expensive numerical simulation or a physical experiment. The major cost of the surrogate model occurs at this stage; that is, each sample requires simulation/experimentation. Once m numbers of samples, $\mathbf{S} = \{S^{(1)}, S^{(2)}, \dots, S^{(m)}\}$, are obtained, Figure 1-3(b) shows a surrogate model that fits using these samples. The surrogate model can be simple polynomials or complicated statistical functions. However, it is assumed that the cost of constructing the surrogate model and evaluating the output QoI using the surrogate model is assumed to be negligible compared to that of evaluating the output QoI samples $y^{(k)}$. Under this assumption, the cost of a surrogate model can be determined based on the number of samples to construct it. In general, the accuracy of a surrogate model is proportional to the number of samples. Therefore, there is a trade-off between the accuracy of a surrogate model and the cost of constructing it.

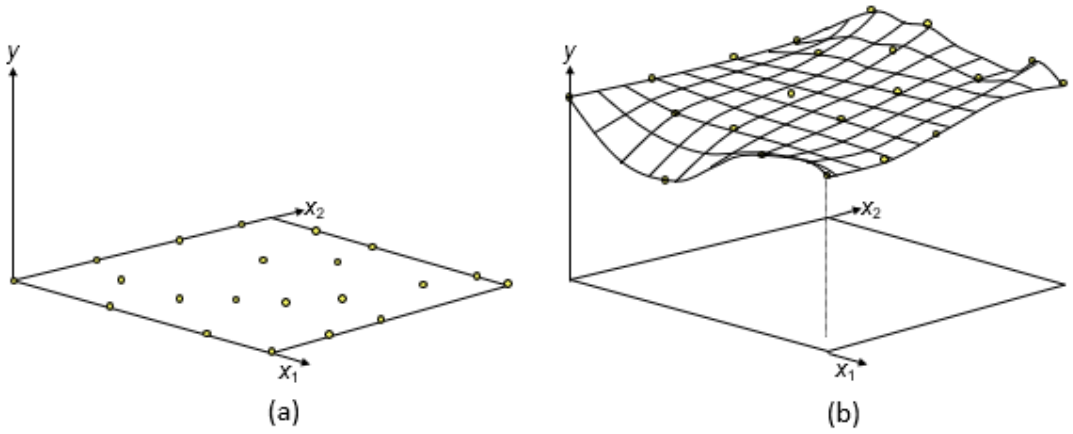


Figure 1-3: An example of samples in a 2D domain and the corresponding surrogate model

Surrogate modeling began in field experiments where the trend of a response change was measured by varying input parameters [4]. Normally, the input parameters were varied by two or three levels, and the change in the response was modeled using linear or quadratic polynomials. Since measured data from an experiment normally contain noise, it was important that the surrogate model filters out the measurement noise. The polynomial response surface (PRS) was designed for this particular purpose, where the functional form was assumed to be a linear combination of monomial bases with unknown

coefficients. These coefficients were determined to minimize the error between the surrogate prediction and measured data. The errors between the surrogate prediction and measured data are considered random noise. Therefore, the surrogate prediction is different from the data due to the presence of noise. The fundamental assumption of PRS is that the form of the surrogate model is correct, but the data have random noise. Figure 1-4(a) shows an example of a 1D PRS surrogate where four samples were used to fit a quadratic function. As expected, the PRS surrogate does not pass through the sample points. Instead, the surrogate prediction distributes positive and negative errors, such that the mean of the errors becomes zero (unbiased). Indeed, this is the process of determining the unknown coefficients of the PRS surrogate. Figure 1-4(a) also shows confidence intervals (green area). We will discuss the PRS surrogate in more detail in Chapter 2.

As computer simulation was greatly advanced due to the development of powerful computers, people started to construct a surrogate model out of computer simulation results rather than physical experiments, which is often called numerical experiments. The data generated from computer simulations have different characteristics from that of physical experiments. Even if a computer simulation has random noise, it is often from numerical truncation error, which is negligible compared to other types of errors. On the other hand, simulation data would not be exact due to various assumptions in modeling and numerical errors. These assumptions and errors tend to generate bias-like predictions; i.e., predictions are shifted from the true data. However, since the goal of surrogate modeling is to approximate simulation results, not the true function, the simulation data are considered exact, and the difference between surrogate prediction and simulation data is considered a prediction error, not noise. Thus, surrogates that predict the simulation data exactly are preferred over ones that cancel noise at data points. For example, the Kriging surrogate model, which is also called the Gaussian process model, is popular for computer simulation. The fundamental assumption of the Kriging surrogate is that data are inherently exact, but the model form has uncertainty. Due to this assumption, the Kriging surrogate predicts the simulation data exactly, but the uncertainty increases as a prediction point is farther away from data points. Figure 1-4(b) shows an example of a Kriging surrogate, where the same four samples were used to fit the surrogate. Since the Kriging surrogate considers that there is uncertainty in the model, the prediction is represented by the mean curve (red-colored curve) with 95% confidence intervals (curves with the green area). Notably, that the prediction uncertainty disappears at sample locations because the Kriging surrogate considers data to be exact; i.e., that there is no uncertainty at data points. We will discuss the Kriging surrogate in more detail in Chapter 7.

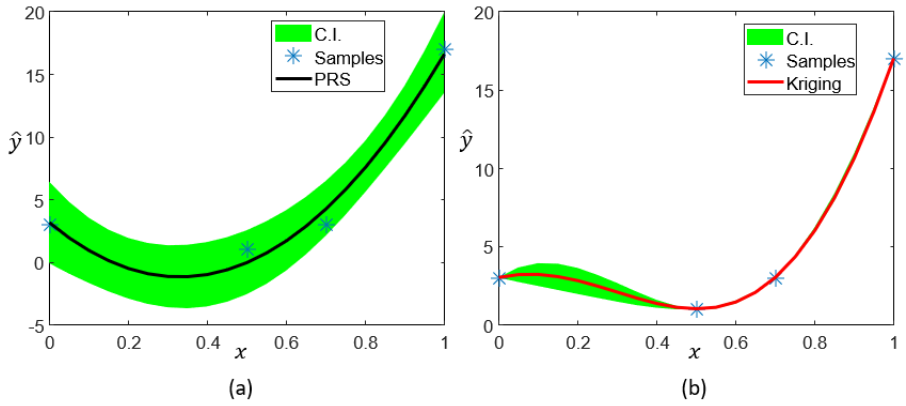


Figure 1-4: Polynomial response surface versus Kriging surrogate models

As discussed in subsequent chapters, there are many different surrogates, including polynomial response surface, Kriging, radial basis function, support vector machine, space mapping, artificial neural network, polynomial chaos expansion, and Bayesian network. Which surrogate model is the best? This is a question without an answer because often the comparison is invalid and meaningless. As discussed previously, the PRS and Kriging surrogates have different assumptions. Therefore, it is obvious that one will be better than the other when its own set of assumptions is satisfied. For example, if the PRS surrogate has the exact model form and the data have Gaussian noise, it may perform better than the Kriging surrogate. On the other hand, if the data are accurate but the PRS surrogate does not have an exact model form, the Kriging surrogate may perform better than the PRS surrogate. When the nature of the true function and its data are unknown, it is not clear which surrogate model will be the most accurate. However, if more information is available, it is possible to find a good surrogate that matches the available information. For example, when the nature of the true function is known, physics-based surrogates such as space-mapping-based models might be most efficient. When a set of data are given, it is difficult to suggest the "best" surrogate. In fact, there is no consensus on how to obtain the most reliable estimates of the accuracy of a given surrogate. Instead of selecting the best surrogate, it might be possible to eliminate bad surrogates. Goel et al. [3] suggested using an ensemble of multiple surrogates to reduce the risk of using a bad surrogate. This is a particularly attractive idea because we assume that the cost of constructing a surrogate is negligible compared to obtaining data. Therefore, it is relatively easy to build multiple surrogates and remove those predictions far away from other surrogates.

1.3. Design of experiments: sampling

As the purpose of surrogate modeling is to approximate a QoI using a set of samples, the goal is to make the surrogate model as accurate as possible using as few samples as possible. To achieve this goal, the first task is to determine how many samples to use and where to locate the samples. The methodology of selecting samples is called "design of experiments (DoE)" or "experimental design". This terminology was developed because PRS was initially designed to approximate responses from experiments. Since this is an important topic in surrogate modeling, Chapter 3 is dedicated to discussing the design of experiments. It is generally accepted that DoE is as important as the surrogate itself because the accuracy of the surrogate depends on the number and location of samples. In addition, there is no single DoE scheme that is the best for all surrogates. Various DoE techniques cater to different sources of errors, in particular, errors due to noise in the data or errors due to an improper surrogate model.

Samples do not need to be generated simultaneously. After starting from a small number of samples, additional samples can be added sequentially to improve the accuracy of surrogate prediction. The strategy of adding additional samples is called adaptive sampling, sequential design, or active learning. Usually, adaptive sampling strategies require a criterion to determine the next sample locations. Many criteria have been proposed, such as D-optimal design and minimum bias design. These criteria are often based on a trade-off between minimizing bias (i.e., error) and/or minimizing variance (i.e., uncertainty).

When a surrogate model is sufficiently flexible, a better surrogate can be obtained with more samples. In addition, it is better to locate the samples such that they are more or less uniformly distributed to the entire sampling space, which is called "space-filling design". To have a space-filling design, some sampling strategies minimize the maximum distance between samples. Others uniformly divide the sampling space and put a sample in each segment.

1.4. Interpolation versus extrapolation

Although it is better to distribute samples uniformly to the entire sampling space, it is often impossible to generate enough samples. This is particularly the case when the dimension of input variables is high. To illustrate, consider two-dimensional space as shown in Figure 1-5(a). First, the range of each input variable is halved so that there are four quadrants. Assume that a sample is located at the center of each quadrant. Therefore, in two-dimensional space, there are 4 samples. Figure 1-5(b) shows the same practice in three dimensions, now with eight samples—one sample at each octant. If the dimension of input variables increases, the number of samples increases rapidly. For example, in the case of ten-dimensional space, 1,024 samples are required to locate a sample to each orthant (orthant is hyperoctant, which is the analog in n-dimensional Euclidean space of an octant in three dimensions). Therefore, the number of samples increases exponentially, proportional to the number of dimensions, which is called the 'curse of dimensionality.'

A more disruptive fact is that one sample at each orthant is not enough for space-filling purposes. to explain, consider the difference between 'interpolation' and 'extrapolation'. Even if there are several different definitions, this book considers that interpolation means that the prediction point is within the convex hull of samples. If the prediction point is out of the convex hull, it is referred to as extrapolation. In general, the prediction in the interpolation region is more accurate than that of the extrapolation region. Therefore, it is natural to generate samples such that they cover the input space as much as possible; i.e., maximizing the interpolation region. The challenge is that in general, the extrapolation region is much larger than the interpolation region. For example, in the two-dimensional case in Figure 1-5(a), the interpolation region is in the grey area, while the extrapolation is outside of that. In the case of a square space, the interpolation region only covers 25% of the entire input space. This portion rapidly decreases as the dimension increases. For example, in the three-dimension in Figure 1-5(b), the interpolation region covers 12.5%, and the 10-dimension covers less than 0.1% of the input space. Therefore, it would be hard to rely on a prediction that comes from 99.9% extrapolation.

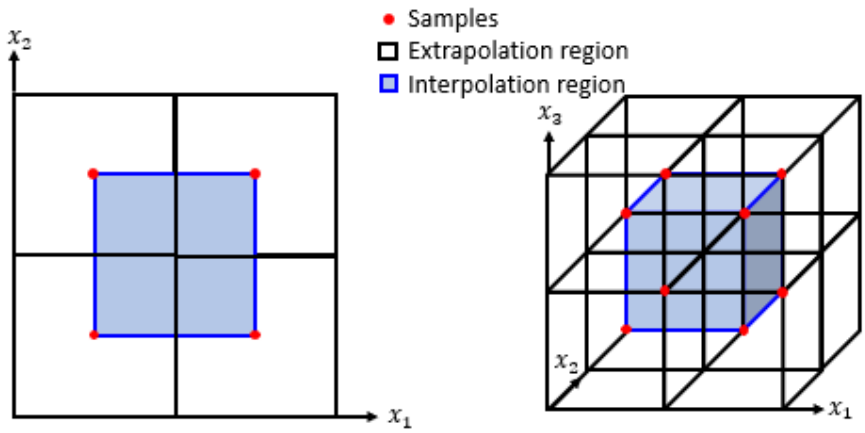


Figure 1-5: Interpolation versus extrapolation region in the sampling space

The issue of extrapolation is a major challenge in surrogate modeling. This is a major problem in both one- and multi-dimensional design space. To illustrate the issue dramatically, consider the following one-dimensional function:

$$y(x) = x + 0.5\sin(5x) \tag{1.2}$$

The function is defined in the domain $x \in [0,10]$. To fit the function, 21 equally-spaced samples were generated in the region $x \in [1,9]$. Therefore, the interpolation region is $x \in [1,9]$, while the extrapolation region is $x \in [0,1] \cup [9,10]$. That is, 80% of the domain is the interpolation region, while 20% is the extrapolation region. The following Matlab code is used to generate samples, fit the surrogate, and plot the surrogate and true functions:

```
x=linspace(1,9,21);
y=x+0.5*sin(5*x);
net=newrb(x,y);
xf=linspace(0,10,101);
yf=xf+0.5*sin(5*xf);
ysim=sim(net,xf);
plot(xf,yf); hold on;
plot(xf,ysim,'ro');
```

For this particular example, the true function in Eq. (1.2) is used to generate samples; that is, there is no error or noise in the samples. Figure 1-6 shows the true function and surrogate prediction along with the interpolation and extrapolation regions. It turns out that the surrogate predictions are accurate in the interpolation region. In fact, the surrogate predictions were exact in this region. In the extrapolation region, however, the surrogate predictions showed significant errors. Considering that the extrapolation region is only 20% of the entire domain and the true function shows a repeated pattern, the surrogate model completely loses its prediction capability.

A surrogate with a good fit in the interpolation region can fail miserably in predicting the extrapolation region. This is one of the biggest challenges associated with surrogate modeling. In general, it is difficult to make the entire input domain the interpolation region, i.e., samples are located in all vertices of the input domain. As shown in Figure 1-5, this will increase the number of samples exponentially, along with the dimension of input parameters. It is also possible that an interpolation region may have significant error if the distance between samples is large. For example, consider the two-dimensional space in Figure 1-5(a). If samples are located at all corners, the maximum distance between samples is $\sqrt{2}$, assuming that all input variables are normalized between $[0, 1]$. This distance increases to $\sqrt{3}$ for three dimensions, and in general, $\sqrt{n}$ for n dimensions. As the distance between the samples increases, the prediction accuracy decreases, which is similar to the behavior seen in the extrapolation region.

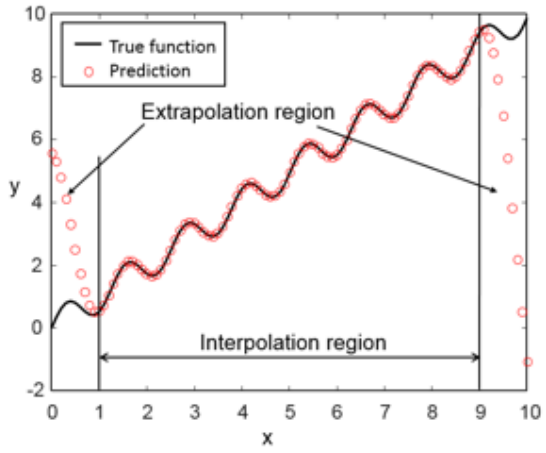


Figure 1-6: Fitting a function using a radial-basis neural network model in Matlab

1.5. Flowchart of surrogate modeling

Since we discussed some important tasks of surrogate modeling individually, it might be beneficial to discuss the flowchart of general surrogate modeling. Figure 1-7 shows a flowchart of surrogate modeling. Although there are many different ways of describing the surrogate modeling process, we use four steps in this text. The first step is the design of experiments (DoE), where the number and location of samples is planned in the input variable space. Normally, the input variables are normalized, and thus, the input variable space is a hyper-cube. Then, DoE distributes samples in this domain. Since there are infinitely many possibilities for locating samples, it is important to assess the goodness of different DoEs. Also, since each sample comes from an expensive computer simulation or experiment, the number of samples is limited.

The second step is to run simulations or experiments at selected DoE locations. As previously mentioned, it is assumed that numerical simulations or experiments are expensive compared to fitting or evaluating the surrogate. When computer simulations are used to evaluate the QoI at sample locations, it is possible that parallel processing can be used to save computational time. It is also noted that the obtained QoI at sample locations might include random noise or error.

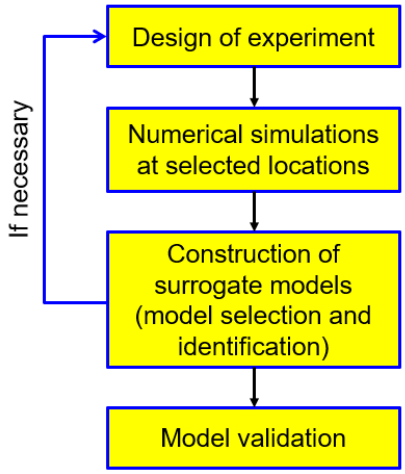


Figure 1-7: Flowchart of surrogate modeling

The third step is to build a surrogate using samples. This step can be broken down into two stages. The first is to select a surrogate model form, and the second is to determine unknown coefficients using samples, which is also called model identification. The model selection is closely related to the number and location of samples. A flexible model usually comes with greater numbers of unknown coefficients, which requires more samples. It is always a good strategy to start with a simple model and progressively increase its complexity. Model identification is the process of determining unknown model parameters by minimizing the errors between the surrogate predictions and samples. Depending on how to define the error, different outcomes can be obtained.

The last step is model validation, which evaluates the accuracy of the constructed surrogate model. There is a philosophical challenge in the validation step. Samples are used to fit the surrogate model, which means the surrogate model is constructed to minimize the errors between the surrogate predictions and samples. Therefore, it is highly likely the errors are small at sample locations. However, this does not mean that the surrogate is a good predictor at un-sampled locations. For example, Figure 1-8 shows a surrogate that passes all sample points but has poor prediction capability. The five samples were generated from a linear function $y = x, x \in [-2, 2]$, while the PRS was assumed to be a quintic

polynomial, which has six unknown coefficients. Since the number of unknown coefficients is greater than the number of samples, the fitting process is indefinite, and it is possible that there are infinitely many combinations of coefficients whose curves pass through all the sample points. Figure 1-8 illustrates one particular case where $y = -2x + 3.75x^3 - 0.75x^5$. In such a case, the surrogate predictions are exact at all sample points. However, the predictions at un-sampled locations are very poor in this example. If only samples are used to evaluate the surrogate, it may decrease the accuracy of the surrogate.

One way of resolving this issue is to save some samples for the purpose of validation. That is, out of m samples, only $k < m$ samples are used for fitting the surrogate, and $m - k$ samples are used for validation. For example, the implementation of the neural network in Matlab uses 70% of the samples for fitting the network while the remaining 30% is used for testing and validation. However, one may get a better surrogate using m samples than using k samples. Therefore, it is good to save $m - k$ samples for the purpose of validation, at the cost of deteriorating the accuracy of the surrogate. One possibility is that $m - 1$ samples are used to fit the surrogate and the other sample is used to measure the prediction error. This process can be repeated m times by dropping one sample at a time to obtain m prediction errors. These m errors can represent the prediction capability of the surrogate. This process is called 'cross-validation.' The advantage of cross-validation is that no samples are wasted for the purpose of validation.

If the model is not sufficiently accurate, it is necessary to add additional samples (adaptive sampling) and reconstruct the surrogate model. If the model form limits the accuracy, it can also be modified. For example, if a linear PRS surrogate cannot represent the trend of data, a higher-order PRS can be used to match the data trend. This remodeling process requires several iterations until the validation error becomes less than a threshold.

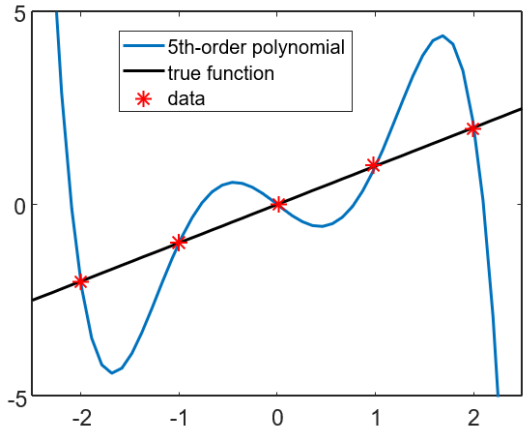


Figure 1-8: A surrogate model that is perfect at sample locations but has a large prediction error at un-sampled locations

1.6. Overview of surrogate modeling

With the flowchart of surrogate modeling in Figure 1-7 in mind, Figure 1-9 shows the overview of surrogate modeling. The simulation-based model, $f(\mathbf{x})$, represents an expensive numerical simulation or physical experiment. Even if the functional form of the simulation model is hidden or it is too complicated to be used, it is possible to evaluate the simulation model at a given sample location $\mathbf{x}_i$, $f_i = f(\mathbf{x}_i)$; this process is referred to as the analysis problem. The outcomes of the analysis problem, $f_i, i = 1, \dots, m$, are called either samples or data. In general, since the analysis problem is considered computationally expensive, it is expected to have a small number of samples. Once m samples are available, they are used to build the estimated model, $\hat{f}(\mathbf{x})$, which is also called an approximate model or a surrogate model. The estimation process can be understood as a nonlinear inverse problem. That is, we

want to determine a continuous function, $\hat{f}(\mathbf{x})$, as a function of input variables from a limited amount of data, $\mathbf{f} = \{f_1, f_2, \dots, f_m\}^T$. For example, the function can be determined by minimizing the error between the data and prediction as

$$\min_{\hat{f} \in H} \hat{f} = \frac{1}{m} \sum_{i=1}^{m} L\left[f_i - \hat{f}(\mathbf{x}_i)\right] \tag{1.3}$$

where $L[\cdot]$ is a loss function used to quantify the empirical error, and H is the family of surrogate models under consideration. The loss function increases if there is a discrepancy between the data and prediction. Different outcomes are expected when different definitions of the loss function are used.

In addition to the nonlinear inverse estimation problem, the data may be exact or may include noise. Therefore, the estimation problem usually does not have a unique solution. That means, it is possible to have multiple estimated models. Such possibility of multiple models is referred to as 'prediction uncertainty' in this text. For example, if the multiple estimated models are represented by a Gaussian distribution, the surrogate prediction can be expressed in terms of the mean and variance. Once the surrogate model is constructed, it is required to evaluate the accuracy of the model, which is called the appraisal problem. The appraisal problem uses several definitions of error, ϵ, to assess the goodness of the estimated model against the simulation-based model. As discussed in Figure 1-7, if the appraisal error is not satisfactory, additional samples are added, or the model form is changed to improve the prediction accuracy.

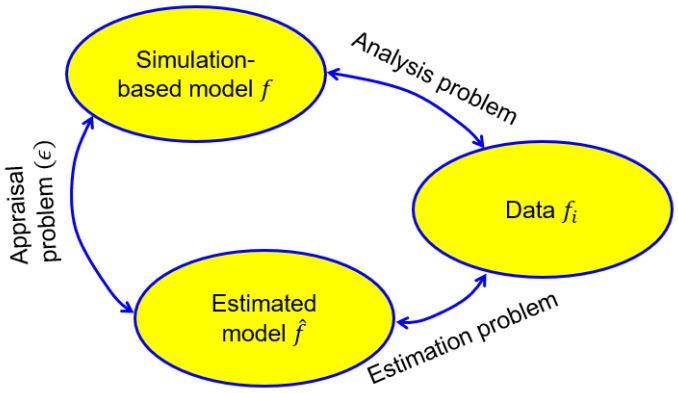


Figure 1-9: Overview of the surrogate modeling process

An introductory level of understanding surrogate modeling would be to check if the surrogate is accurate or not at sample locations. The next level would be to understand the accuracy at un-sampled points (i.e., prediction points). The accuracy at prediction points indeed represents the prediction capability of the surrogate. In order to assess the accuracy at prediction points, the concept of 'prediction uncertainty' is often used. Consider the one-dimensional surrogate model (blue-colored curve) that fits five samples in Figure 1-10. The surrogate has interpolation capability such that the surrogate passes through all sample points; that is, the prediction error is zero at sample locations. However, there is no proof that the surrogate is accurate at un-sampled locations. Therefore, it is natural to assume that there is a prediction error at un-sampled locations. A rigorous method of expressing surrogate prediction can be written as

$$f_p(\mathbf{x}) = \hat{f}(\mathbf{x}) + \epsilon(\mathbf{x}) \tag{1.4}$$

where $\hat{f}(\mathbf{x})$ is the surrogate model, and $\epsilon(\mathbf{x})$ is the prediction error. Some surrogate models only provide information regarding $\hat{f}(\mathbf{x})$, but other surrogates provide additional information regarding $\epsilon(\mathbf{x})$. For example, both PRS and Kriging surrogates assume that the surrogate prediction is uncertain with $\hat{f}(\mathbf{x})$ being the mean prediction. The uncertainty presents in the prediction error, which is often represented by a normal distribution with zero mean:

$$\epsilon(\mathbf{x}) \sim N\left(0, \sigma^2(\mathbf{x})\right) \tag{1.5}$$

where $\sigma^2(\mathbf{x})$ is called the prediction variance. In this perspective, the surrogate prediction is a distribution with the expected value $E(f_p) = \hat{f}(\mathbf{x})$ and variance $V(f_p) = \sigma^2(\mathbf{x})$.

Considering the mean and variance of surrogate prediction, the quality of a surrogate can be evaluated in two different aspects. The first is based on the accuracy of the mean prediction, while the second is based on the prediction uncertainty. Even if a mean prediction shows a good accuracy compared with samples, a large prediction uncertainty may indicate that it is possible that the true QoI can be far away from the mean prediction. Therefore, a good surrogate should have a good mean prediction as well as a small prediction uncertainty.

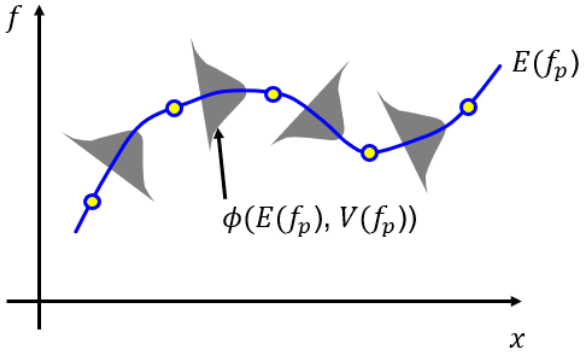


Figure 1-10: Prediction uncertainty in a surrogate

1.7. Smoothness and loss function

As mentioned previously, surrogate modeling can be considered a nonlinear inverse problem where a continuous function is identified by using a finite number of samples. Since the inverse problem does not have a unique solution, it is possible to have infinitely many surrogates that satisfy the inverse problem. For example, Figure 1-11 shows that two surrogate models pass through the five samples. Since both surrogates pass through the samples, the fitting errors of both surrogates are zero. Therefore, it is unclear which surrogate is better.

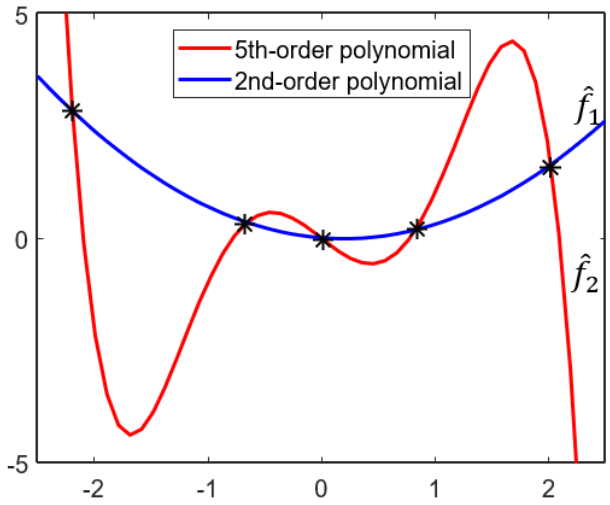


Figure 1-11: Different surrogate that fits the same samples

If multiple surrogates have the same fitting error, a smoother one is considered a better one. In the case of the two surrogates in Figure 1-11, the blue one is considered to be smoother than the red one. Although both surrogates are a smooth function of input variable x, the blue surrogate changes the sign of the slope less than that of the red one. Therefore, the smoothness is related to the change in the sign of the slope; i.e., gradient. When a PRS surrogate is used, a surrogate with lower polynomial orders is considered smoother than the one with higher polynomial orders. For example, a cubic polynomial function has one inflection point, while a quartic polynomial function has two. Therefore, a polynomial function with a less number of inflection points is considered smoother.

The nonlinear inverse estimation problem in Eq. (1.3) can be modified such that a smoother model can be preferred. Knowing that the smoothness is related to the change in the sign of the slope, a function with high-order derivatives can be penalized during the estimation process. That is, the following formulation can be used instead of Eq. (1.3):

$$\min_{\hat{f}\in H} \hat{f} = \frac{1}{m}\sum_{i=1}^{m} L\left[f_i - \hat{f}(\mathbf{x}_i)\right] + \lambda \int \left\| D^k \hat{f} \right\|_H \mathrm{d}\mathbf{x} \tag{1.6}$$

where the error is represented in the loss function, while the smoothness is represented using $D^k\hat{f}$, which is kth-order derivative of $\hat{f}$. When a surrogate has higher-order derivatives, it is penalized by the regularization parameter $\lambda \geq 0$. With a large λ, the estimation process will choose a smooth function even if the function has a relatively large error, which causes a biased error. With a small λ, the process will choose the most flexible model to fit the data.

In Eqs. (1.3) and (1.6), the loss function $L[\cdot]$ is used to determine the closedness of the model to the data. The loss function is supposed to increase as the error term, $e_i = f_i - \hat{f}(\mathbf{x}_i)$, increases. There are many different forms of a loss function. Figure 1-12 shows examples of loss functions. First, the quadratic loss function penalizes a large error using a quadratic function. This is equivalent to the L_2-norm in mathematics, the square sum of errors. This is the most commonly used loss function as it is a smooth function of error. The quadratic loss function is easy to estimate its parameters because its derivatives become a linear system of equations. However, this function is sensitive to a small number of outliers (i.e., large errors), and tends to ignore the effect of small errors. Second, the linear loss function is also called the Laplace loss function, which is equivalent to L_1-norm in mathematics. This basically uses the absolute value of the errors such that the contribution is linearly proportional to the magnitude. The linear

loss function is not smooth at the origin. Third, the Huber loss function in Figure 1-12(c) tries to take advantage of both the linear and quadratic loss functions. It is quadratic for small errors while linear for large ones. Therefore, the Huber loss function is smooth at the origin and not too sensitive to large errors. Last, the ϵ-loss function in Figure 1-12(d) is popular in support vector regress surrogates, where errors whose magnitude is less than ϵ are considered to be zero. This loss function is particularly useful when data include small random noise, which should not affect the decision-making process for selecting a surrogate.

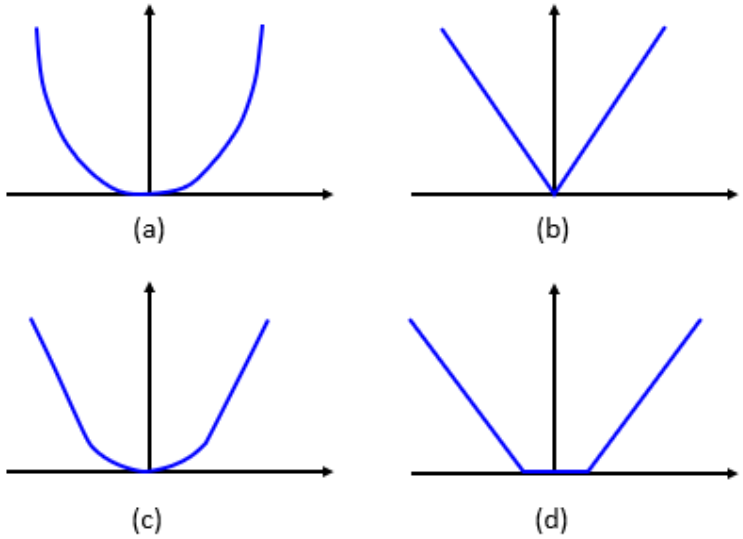


Figure 1-12: Loss functions, (a) quadratic loss function (L_2-norm), (b) linear loss function (L_1-norm), (c) Huber loss function, (d) ϵ-loss function