The dual representation and its application to seismic reservoir analysis

Brian Russell

Banff, December 2, 2022

NSERC CRSNG
UNIVERSITY OF CALGARY
FACULTY OF SCIENCE
Department of Geoscience
GeoSoftware™

Introduction

- Statistical and machine learning techniques can be broken into two broad classes:
  - Linear regression and classification techniques.
  - Nonlinear regression and classification techniques.
- In this talk, I will focus on the relationship between linear and nonlinear regression techniques.
- In geophysics, we are familiar with the primal least-squares approach, and I will first review that method.
- I will then introduce the dual representation, which can be derived from the primal method with a pre-whitening term.
- I will show that nonlinear regression techniques can then be derived using the kernel substitution approach, which is based on the dual representation.
- I will show several simple model examples and then finish with a case study from the Blackfoot dataset.

The linear model

- In geophysics and machine learning, the linear model is written:

$$\boldsymbol{y} = X\boldsymbol{w}, \text{ where:}$$

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}, X = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1D} \\ 1 & x_{21} & \cdots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N1} & \cdots & x_{ND} \end{bmatrix}, \text{ and } \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_D \end{bmatrix}.$$

- In the above model, $\boldsymbol{y}$ is a vector of N observations, X is matrix consisting of D columns of attributes, or features, plus a column of ones, and $\boldsymbol{w}$ is a vector consisting of $D+1$ weights, where w_0 is called the bias.
- Examples of this model include AVO analysis, predicting a reservoir parameter from multiple attributes, and seismic tomography.

The linear basis function model

- The linear basis function model is an extension of a linear model where $\boldsymbol{x}$ is replaced with $M+1$ nonlinear basis functions $\phi_j(\boldsymbol{x})$, or:

$$y_i = w_0\phi_0(\boldsymbol{x}_i) + w_1\phi_1(\boldsymbol{x}_i) + \ldots + w_M\phi_M(\boldsymbol{x}_i), \;\; i = 1, \ldots, N$$

- As with the linear model, the linear basis function model can be written in matrix format as follows, but note that this is now an N x $M+1$ dimensional matrix:

$$\boldsymbol{y} = \Phi\boldsymbol{w}, \text{ where}$$

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}, \; \Phi = \begin{bmatrix} \phi_0(\boldsymbol{x}_1) & \phi_1(\boldsymbol{x}_1) & \ldots & \phi_M(\boldsymbol{x}_1) \\ \phi_0(\boldsymbol{x}_2) & \phi_1(\boldsymbol{x}_2) & \ldots & \phi_M(\boldsymbol{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\boldsymbol{x}_N) & \phi_1(\boldsymbol{x}_N) & \ldots & \phi_M(\boldsymbol{x}_N) \end{bmatrix}, \text{ and } \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_M \end{bmatrix}$$

Basis functions

- The two most common basis functions are the polynomial and the Gaussian (a third is the sigmoidal function, but I will only discuss the first two today).
- If we consider the case of $D = 1$, the polynomial function is written:

$$\phi_j(x_i) = x_i^j, j = 0,1,\ldots,M$$

- If we substitute the polynomial function into the previous equation, we get:

$$y_i = w_0 + w_1 x_i + w_2 x_i^2 + \ldots + w_M x_i^M$$

- The Gaussian function is written:

$$\phi_j(x_i) = \exp\left[-\frac{(x_i - \mu_j)^2}{2\sigma^2}\right], \text{where } \mu_j = \text{mean values, and } \sigma^2 = \text{variance.}$$

- If we substitute the Gaussian function into the previous equation, we get:

$$y_i = w_0 \exp\left[-\frac{(x_i - \mu_0)^2}{2\sigma^2}\right] + \ldots + w_M \exp\left[-\frac{(x_i - \mu_M)^2}{2\sigma^2}\right]$$

Solving for the weights – the primal and dual solutions

- You will be familiar with the primal least-squares solution given by:

$$\boldsymbol{w} = \left(\Phi^T \Phi + \lambda I_{M+1}\right)^{-1} \Phi^T \boldsymbol{y},$$

- where $\Phi^T\Phi$ is the inner product of Φ, λ is the pre-whitening term and I_{M+1} is the M+1 x M+1 identity matrix.
- The pre-whitening form of the inverse also allows us to derive the dual form of the solution (see Appendix), given by:

$$\boldsymbol{w} = \Phi^T \left(\Phi\Phi^T + \lambda I_N\right)^{-1} \boldsymbol{y},$$

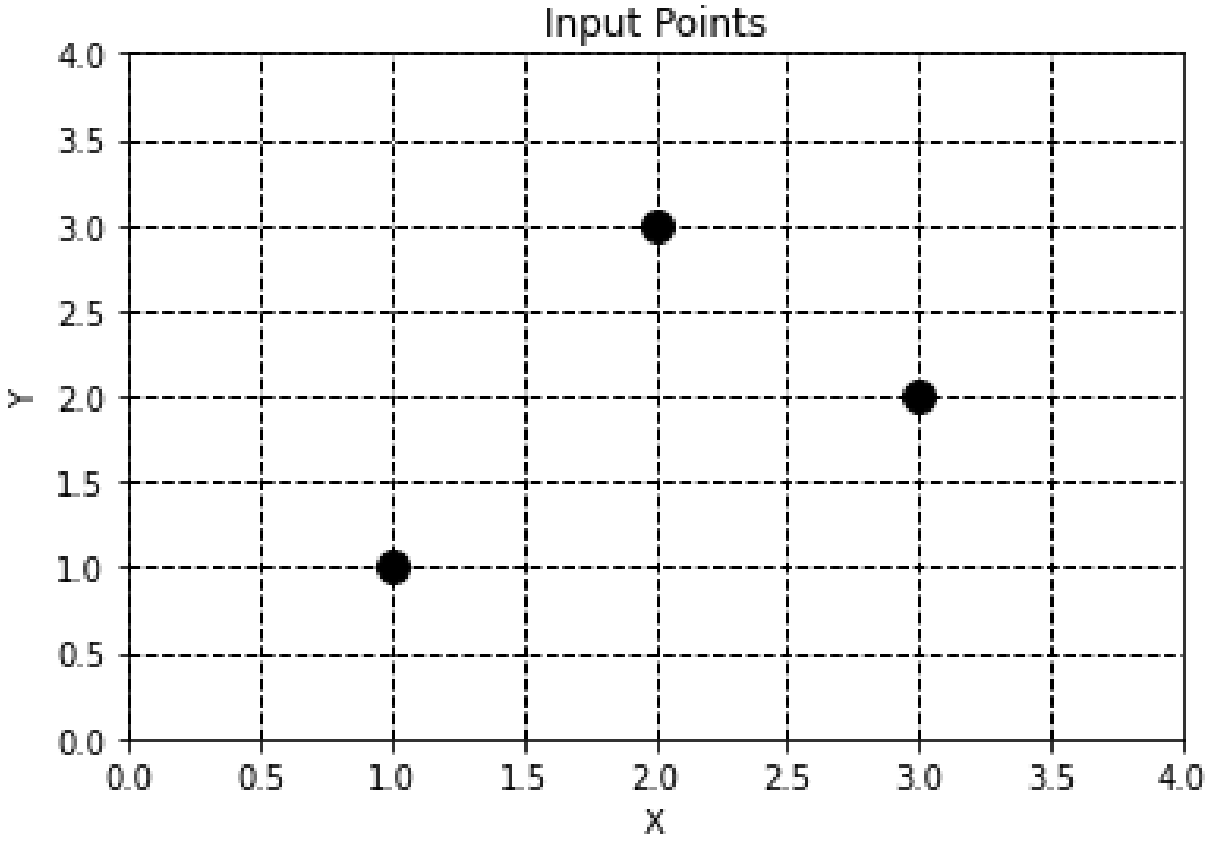
- where $\Phi\Phi^T$ is the outer product of Φ, an N x N square matrix, and λI_N is a pre-whitening term that now uses the N x N identity matrix.
- Note that we can introduce an M+1 x N size matrix $\Phi^\dagger$, called the generalized inverse, which is identical for both formulations:

$$\Phi^\dagger = \left(\Phi^T \Phi + \lambda I_{M+1}\right)^{-1} \Phi^T = \Phi^T \left(\Phi\Phi^T + \lambda I_N\right)^{-1}$$

Linear polynomial regression

- The simplest example has $D = 1$, $M = 1$, and $N = 3$ points:

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \text{ and } \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix}$$



- This gives a set of 3 equations with 2 unknowns:

$$y_1 = w_0 + w_1 x_1$$
$$y_2 = w_0 + w_1 x_2$$
$$y_3 = w_0 + w_1 x_3$$

- Since $N > M + 1$, this is called the over-determined case, since there are more observations than unknowns.

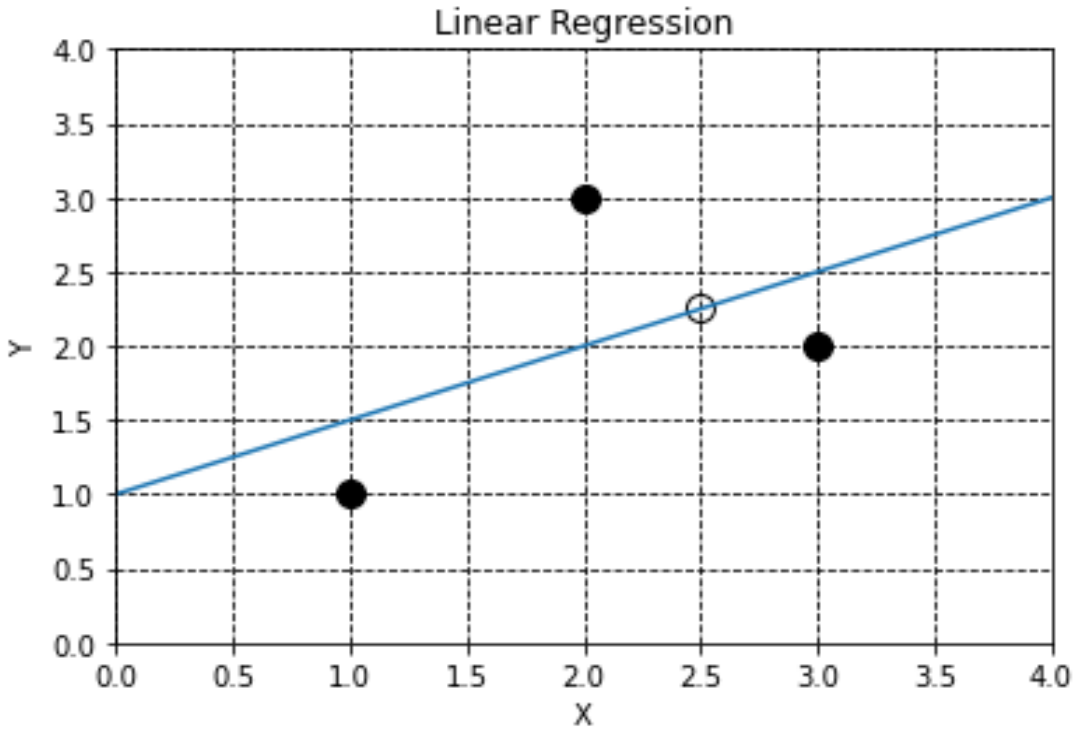
Primal and dual least-squares solution

- These equations can be written in matrix form as:

$$\boldsymbol{y} = \Phi \boldsymbol{w}, \text{ where } \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}, \Phi = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}.$$

- Using $\lambda = 0$ for the primal solution and 10^{-6} for the dual solution we get the generalized inverse solution below and the regression line shown on the right:

$$\boldsymbol{w} = \begin{bmatrix} 4/3 & 1/3 & -2/3 \\ -1/2 & 0 & 1/2 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$$



- The linear regression line can be written for each point x as:

$$\hat{y}(x) = \begin{bmatrix} 1 & x \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = w_0 + w_1 x$$

- For the open circle, this gives:

$$\hat{y}(2.5) = 1 + 0.5(2.5) = 2.25$$

The kernel matrix

- It is obvious that inverting the 2 x 2 matrix in the primal form is more efficient than inverting the 3 x 3 matrix in the dual form.
- But if we write the outer product matrix in its expanded form, we find that:

$$\Phi\Phi^T = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \end{bmatrix} = \begin{bmatrix} 1+x_1^2 & 1+x_1x_2 & 1+x_1x_3 \\ 1+x_2x_1 & 1+x_2^2 & 1+x_2x_3 \\ 1+x_3x_1 & 1+x_3x_2 & 1+x_3^2 \end{bmatrix}$$

- Notice that this means that the matrix multiplication can be written in an alternate way, where each term is computed analytically:

$$K_{ij} = \Phi\Phi^T = 1 + x_i x_j$$

- This is called the kernel matrix and is the fundamental idea of this talk.
- Before looking at the general case, let's go to quadratic polynomial regression.

Quadratic regression

- If we let $p = 2$ in the polynomial equation, we get 3 equations with 3 unknowns:

$$y_1 = w_0 + w_1 x_1 + w_2 x_1^2$$
$$y_2 = w_0 + w_1 x_2 + w_2 x_2^2$$
$$y_3 = w_0 + w_1 x_3 + w_2 x_3^2$$

- These three quadratic equations can be put into the following matrix form, where the third column of X is equal to the square of the $\boldsymbol{x}$ values:

$$\boldsymbol{y} = \Phi \boldsymbol{w}, \text{ where } \Phi = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{bmatrix}, \text{ and } \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}.$$

- Since Φ is now square it can be inverted directly, and both the primal and dual solutions will give the same answer without any pre-whitening.

The quadratic regression fit

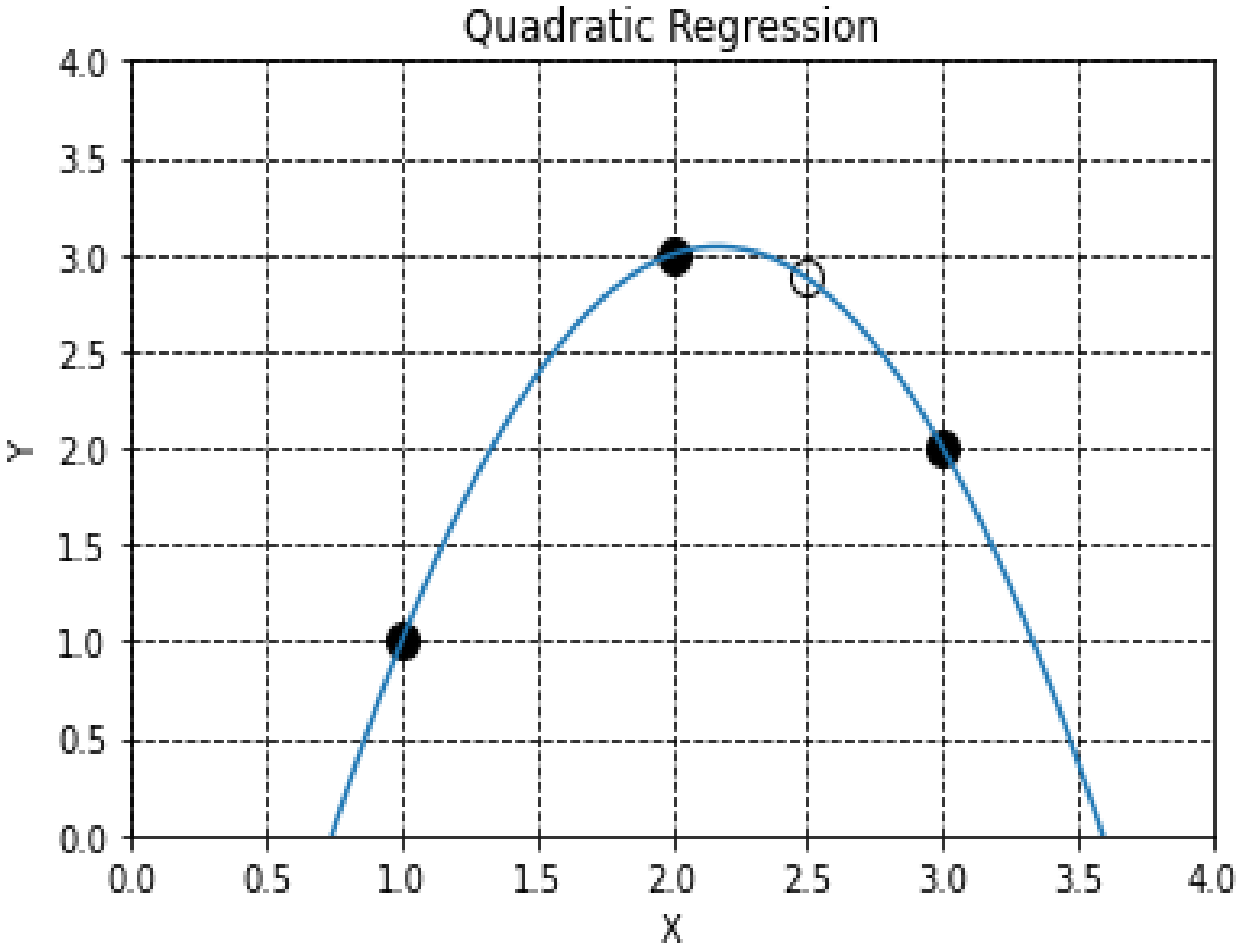
- The quadratic fit to our points can therefore be computed as follows:

$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} -4 \\ 6.5 \\ -1.5 \end{bmatrix}$$

- The resulting fit is shown on the right and is as follows:

$$\hat{y}(x) = -4 + 6.5x - 1.5x^2 \Rightarrow$$

$$\hat{y}(2.5) = -4 + 6.5(2.5) - 1.5(6.25) = 2.875$$



- Notice the perfect fit, but how rapidly it becomes negative to the left and right of the points, unlike the linear fit.

The dual product in quadratic regression

- The outer product matrix for the quadratic case is:

$$\Phi\Phi^T = \begin{bmatrix} 1+x_1^2+x_1^4 & 1+x_1x_2+x_1^2x_2^2 & 1+x_1x_3+x_1^2x_3^2 \\ 1+x_2x_1+x_2^2x_1^2 & 1+x_2^2+x_2^4 & 1+x_2x_3+x_2^2x_3^2 \\ 1+x_3x_1+x_3^2x_1^2 & 1+x_3x_2+x_3^2x_2^2 & 1+x_3^2+x_3^4 \end{bmatrix}$$

- This suggests that the kernel matrix be written as follows, which is very close to the correct answer except for one extra x_ix_j term:

$$K_{ij} = \left(1+x_ix_j\right)^2 \approx \Phi\Phi^T, \text{ since} \left(1+x_ix_j\right)^2 = 1+2x_ix_j+x_i^2x_j^2$$

- Before leaving this simple problem, let's look at the cubic polynomial, since it will teach us the important concept of over-fitting.

Cubic regression

- If we let $M = 3$ in the polynomial equation, we get 3 equations with 4 unknowns, which is the underdetermined case since we have more unknowns than observations:

$$y_1 = w_0 + w_1 x_1 + w_2 x_1^2 + w_3 x_1^3$$
$$y_2 = w_0 + w_1 x_2 + w_2 x_2^2 + w_3 x_2^3$$
$$y_3 = w_0 + w_1 x_3 + w_2 x_3^2 + w_3 x_3^3$$

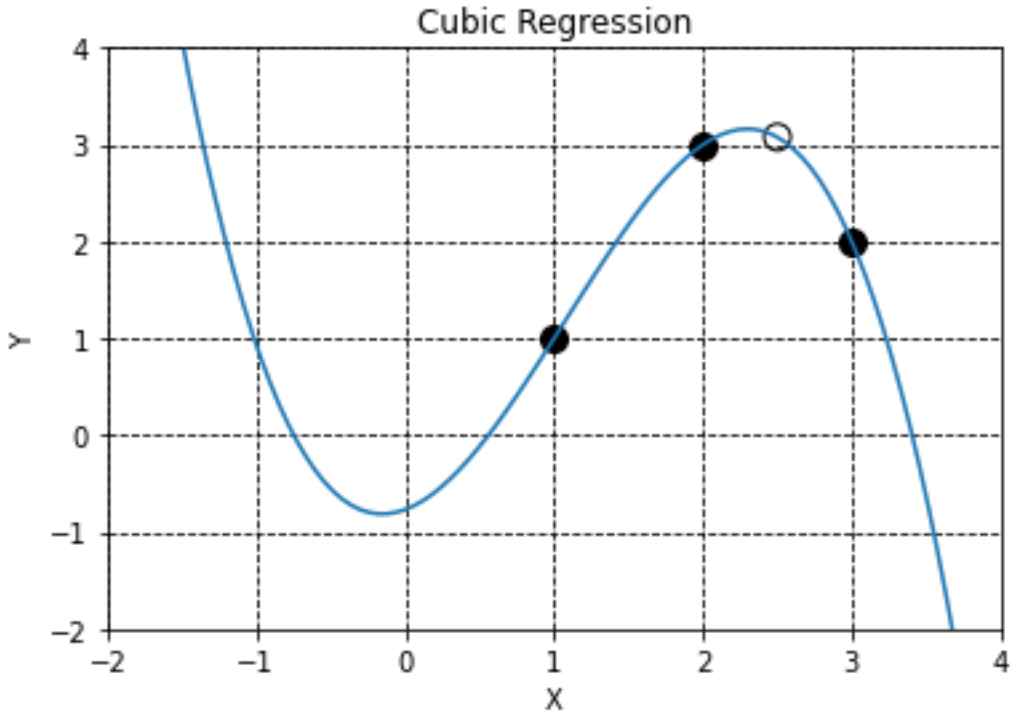
- These three linear equations can be put into the following matrix form, where the fourth column of Φ is equal to the cube of the $\boldsymbol{x}$ values:

$$\boldsymbol{y} = \Phi \boldsymbol{w}, \text{ where } \Phi = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \end{bmatrix}, \text{ and } \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix}.$$

Cubic regression solution

- For the cubic solution, the primal form now needs pre-whitening, whereas the dual form gives the correct answer without pre-whitening:

$$\boldsymbol{w} = \Phi^T \left(\Phi^T \Phi\right)^{-1} \boldsymbol{y} = \begin{bmatrix} -0.768 \\ 0.575 \\ 1.732 \\ -0.539 \end{bmatrix}$$



- The resulting fit is shown on the right and is as follows:

$$\hat{y}(x) = -0.768 + 0.575x + 1.732x^2 - 0.539x^3 \Rightarrow$$

$$\hat{y}(2.5) = -0.768 + 0.575(2.5) + 1.732(6.25) - 0.539(15.625)$$

$$\Rightarrow \hat{y}(2.5) = 3.077$$

- However, since this problem is over-determined, the solution overfits the problem with an extra “swing”.

The dual solution to cubic regression

- Now let's write the outer product matrix for the cubic regression problem:

$$\Phi\Phi^T = \begin{bmatrix} 1+x_1^2+x_1^4+x_1^6 & 1+x_1x_2+x_1^2x_2^2+x_1^3x_2^3 & 1+x_1x_3+x_1^2x_3^2+x_1^3x_3^3 \\ 1+x_2x_1+x_2^2x_1^2+x_2^3x_1^3 & 1+x_2^2+x_2^4+x_2^6 & 1+x_2x_3+x_2^2x_3^2+x_2^3x_3^3 \\ 1+x_3x_1+x_3^2x_1^2+x_3^3x_1^3 & 1+x_3x_2+x_3^2x_2^2+x_3^3x_2^3 & 1+x_3^2+x_3^4+x_3^6 \end{bmatrix}$$

- This suggests that the kernel matrix be written approximately as follows, which has two extra x_ix_j and $x_i^2x_j^2$ terms from the matrix multiplication:

$$K_{ij} = \left(1+x_ix_j\right)^3 \approx \Phi\Phi^T, \text{ since} \left(1+x_ix_j\right)^3 = 1+3x_ix_j+3x_i^2x_j^2+x_i^3x_j^3$$

- I think you can see a pattern here as we move to higher polynomials!
- Next, let's look at the general theory of kernels.

Kernel functions

- What I have been leading to with the simple three-point polynomial regression example is that we can re-write the dual formulation as follows:

$$\boldsymbol{w} = \Phi^T \left(K + \lambda I_N \right)^{-1} \boldsymbol{y}, \text{ where } K \text{ is called the kernel matrix.}$$

- By using what is called kernel substitution, or the kernel "trick", we can obtain the kernel matrix K as a function of the x_i and x_j terms, which for the M^{th} order polynomial can be written as follows:

$$K_{ij} = k(x_i, x_j) = (1 + x_i x_j)^M$$

- Another way to write the above equation is as follows:

$$\boldsymbol{w} = \Phi^T \boldsymbol{a}, \text{ where } \boldsymbol{a} = \left(K + \lambda I_N \right)^{-1} \boldsymbol{y} = \begin{bmatrix} a_1 & \cdots & a_N \end{bmatrix}^T \text{ are called the dual variables.}$$

Applying the kernel function

- The kernel substitution trick allows us to compute the kernel matrix without having to know the matrix Φ.
- But how do we deal with the term Φ in the computation of the weights?
- To understand this, note that we are only interested in the application of the weights, so we can compute the output as:

$$\hat{y}(x) = k(x, x_i)\boldsymbol{a}$$

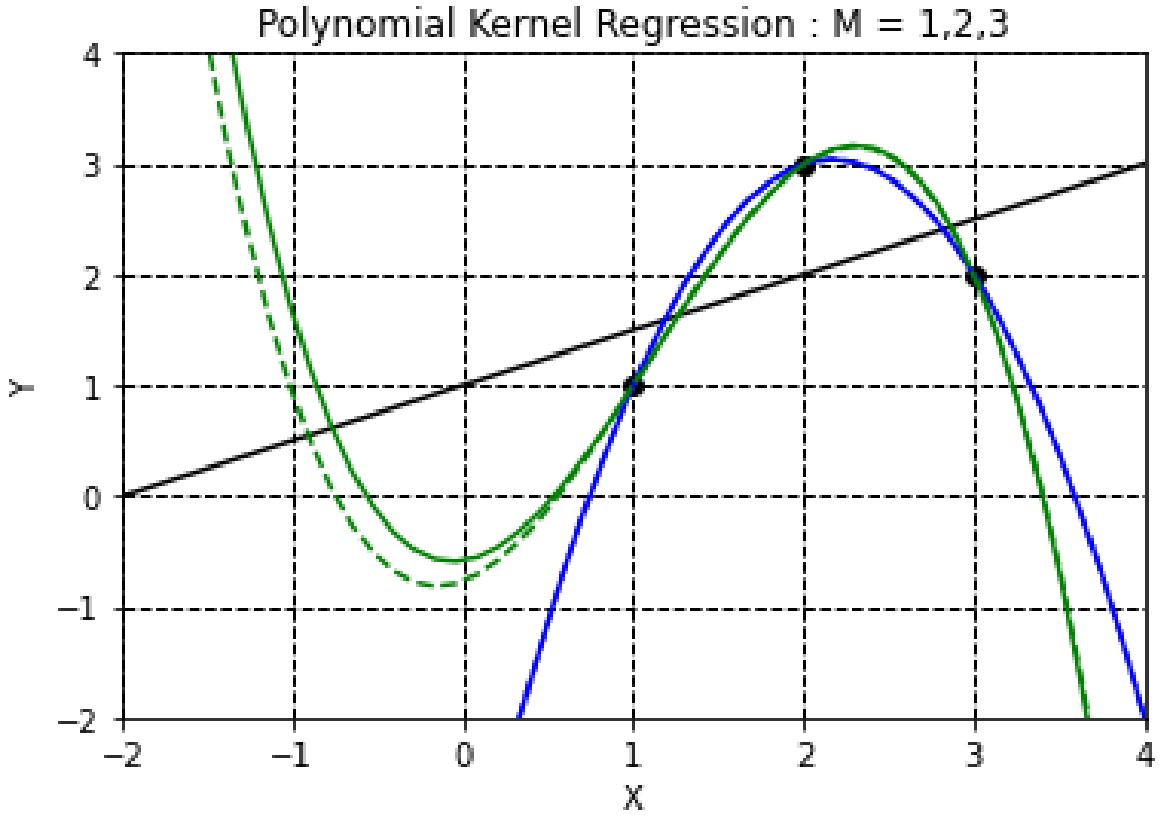
- That is, the output at each sample x simply applies the kernel function between the output sample and each training sample.
- For example, with the polynomial kernel we get:

$$\hat{y}(x) = a_1(1 + xx_1)^M + \ldots + a_N(1 + xx_N)^M$$

- The results of applying this method to our three-point problem is shown in the next slide.

Polynomial kernels

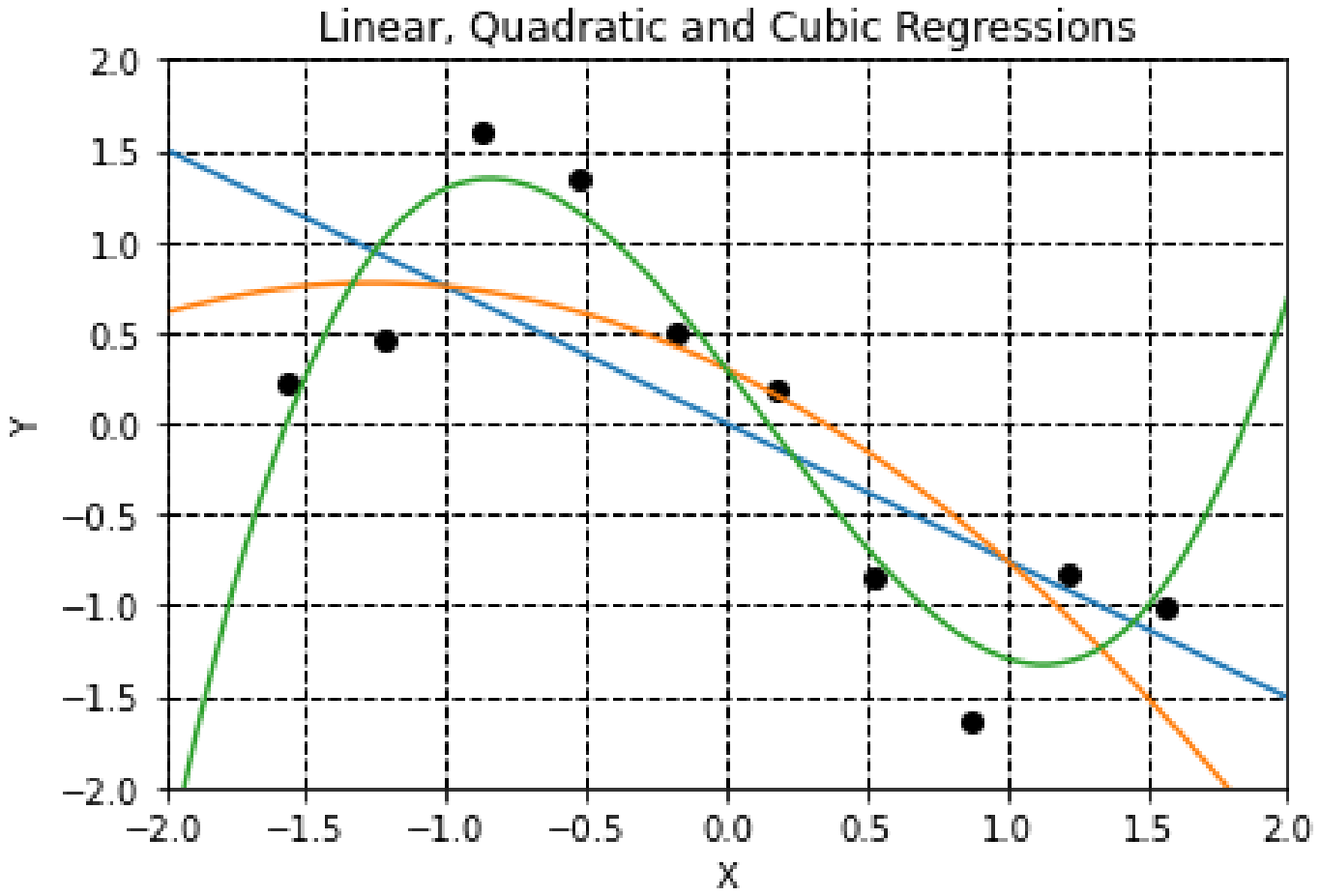
- Linear, quadratic, and cubic regression fits for the 3-point example, where the solid lines use the matrix approach and the dashed lines use the kernel approach,
- Note that the linear fit is perfect, the quadratic fit is close to perfect, but the cubic fit deviates as we move left.
- Obviously, going to higher order polynomials does not make sense for the three-point problem.



- So let's now looks at a more complex ten-point noisy sine wave example, which will show the full advantage of using the kernel method.

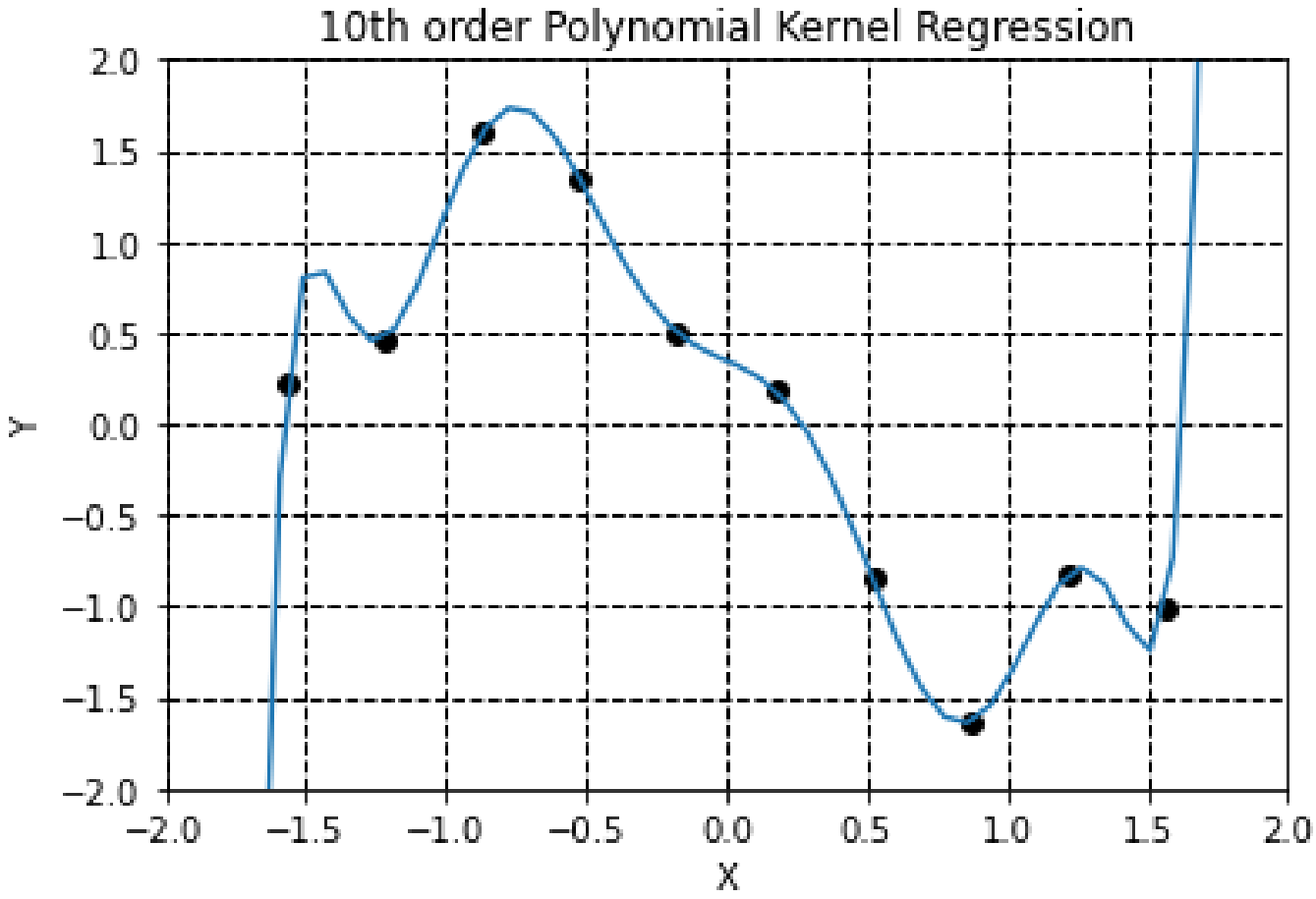
Noisy sine wave polynomial fits

- Here are the linear, quadratic and cubic polynomial kernel regression fits for a noisy sine wave example.
- The fits are close to those done using matrix polynomial regression.
- However, as seen on the next slide, using the kernel method makes it much easier to compute higher order polynomial fits.



Noisy sine wave 10th order polynomial kernel fit

- Here is the polynomial kernel regression fit for the noisy sine wave example using $M = 10$.
- Since $M = N$, the number of points, the fit is now perfect for each point.
- However, note the wild downward and upward swings at each end of the fit, which means that we have only done an exact fit within the range of points shown here.
- This is therefore an example of overfitting.



The Gaussian kernel

- Next, let's move to the Gaussian kernel, which can be written as:

$$K_{ij} = \exp\left[-\frac{(x_i - x_j)^2}{2\sigma^2}\right]$$

- For our 3-point problem, this can be written out in full as:

$$K = \begin{bmatrix} 1 & \exp\left[-\frac{(x_1 - x_2)^2}{2\sigma^2}\right] & \exp\left[-\frac{(x_1 - x_3)^2}{2\sigma^2}\right] \\ \exp\left[-\frac{(x_2 - x_1)^2}{2\sigma^2}\right] & 1 & \exp\left[-\frac{(x_2 - x_3)^2}{2\sigma^2}\right] \\ \exp\left[-\frac{(x_3 - x_1)^2}{2\sigma^2}\right] & \exp\left[-\frac{(x_3 - x_2)^2}{2\sigma^2}\right] & 1 \end{bmatrix}, \text{ since } \exp\left[-\frac{(x_i - x_i)^2}{2\sigma^2}\right] = 1$$

The Gaussian kernel

- The results of the Gaussian kernel inversion are applied to predict the unknown from the known points as follows:

$$\hat{y}(x) = a_1\phi_1 + a_2\phi_2 + \ldots + a_N\phi_N, \text{ where}$$

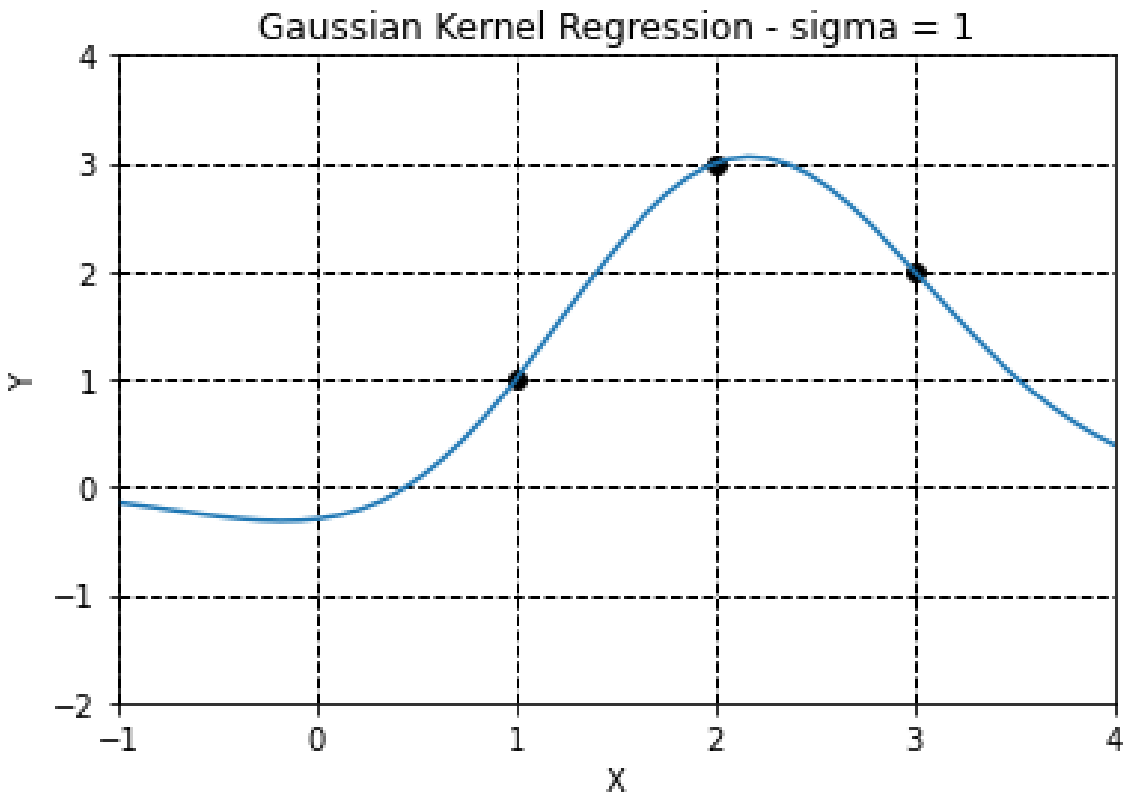
$$\phi_i = \exp\left[-\frac{(x - x_i)^2}{2\sigma^2}\right], i = 1, \ldots, N, \text{ and } \boldsymbol{a} = \left(K + \lambda I_N\right)^{-1} \boldsymbol{y}.$$

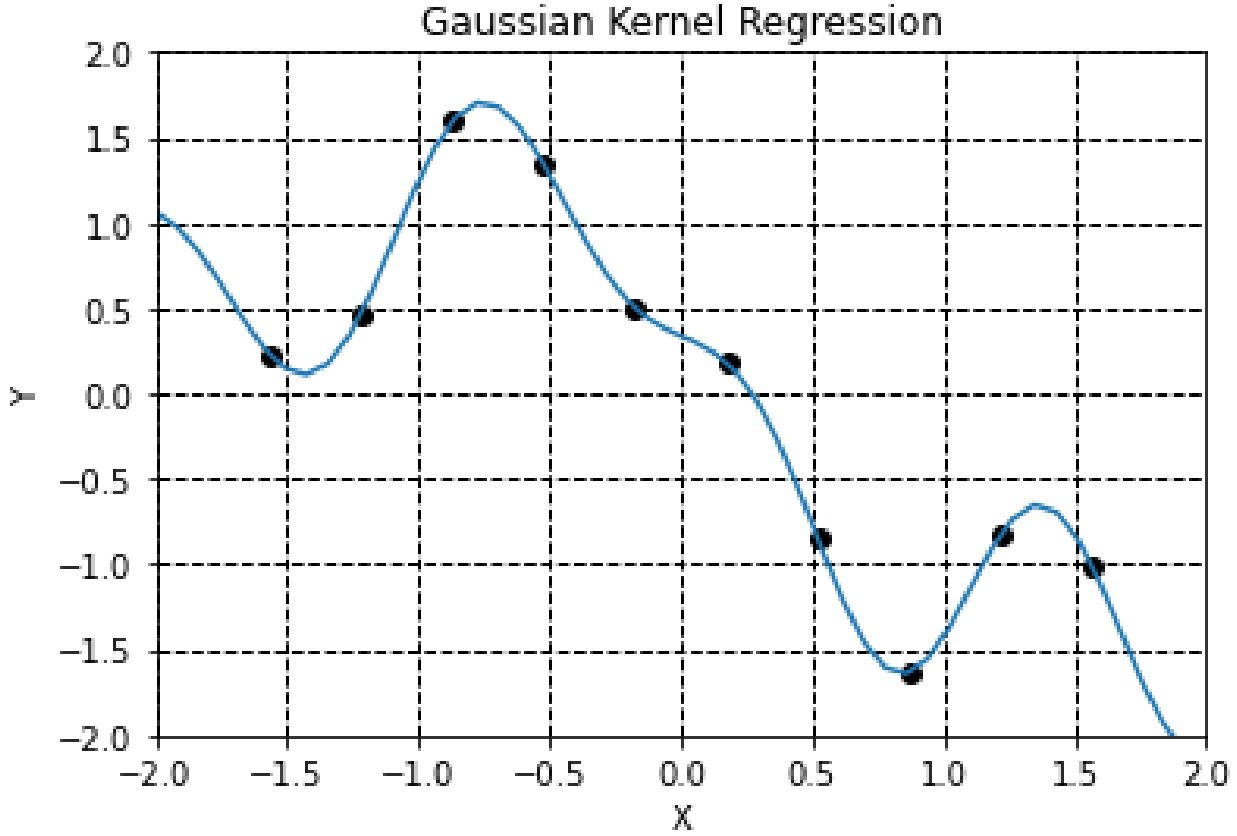
- For the 3-point problem, this can be written as follows for each computed value:

$$\hat{y}(x) = a_1 \exp\left[-\frac{(x_1 - x)^2}{2\sigma^2}\right] + a_2 \exp\left[-\frac{(x_2 - x)^2}{2\sigma^2}\right] + a_3 \exp\left[-\frac{(x_3 - x)^2}{2\sigma^2}\right]$$

- The results of applying the Gaussian kernel to our three-point and noisy sine wave examples are shown on the next slide.

Gaussian kernel regression for 3 and 10-point examples





- Here is the result of applying Gaussian Kernel regression to our three and ten-point examples using $\sigma = 1.0$.
- Notice that we get a perfect fit to the points and that the values trend quite well beyond the first and last points.

The Radial Basis Function Neural Network (RBFN)

- The Gaussian kernel method is also a type of Radial Basis Function Neural Network (RBFN).
- Radial basis functions were introduced by Powell (1987) to perform exact function interpolation and were then recognized as a type of regression neural network.
- They were first introduced into seismic reservoir prediction by Ronen et al. (1994).
- They are called radial basis functions because each function depends only on the radial distance from a center μ, or:

$$\phi_i(x) = h\left(\left\|x - \mu_i\right\|\right), \text{where } h = \text{ a radial basis function.}$$

- When the number of radial basis functions equals the number of points, we get full interpolation:

$$f(x) = \sum_{i=1}^{N} w_i h\left(\left\|x - x_i\right\|\right)$$

- There are many types of radial basis functions, but the most common one is the Gaussian, which is identical to the Gaussian kernel method.

The Generalized Regression Neural Network

- The Generalized Regression Neural Network (GRNN), is like the Gaussian kernel method except than no matrix inversion is involved.
- Recall that Gaussian kernel regression was written:

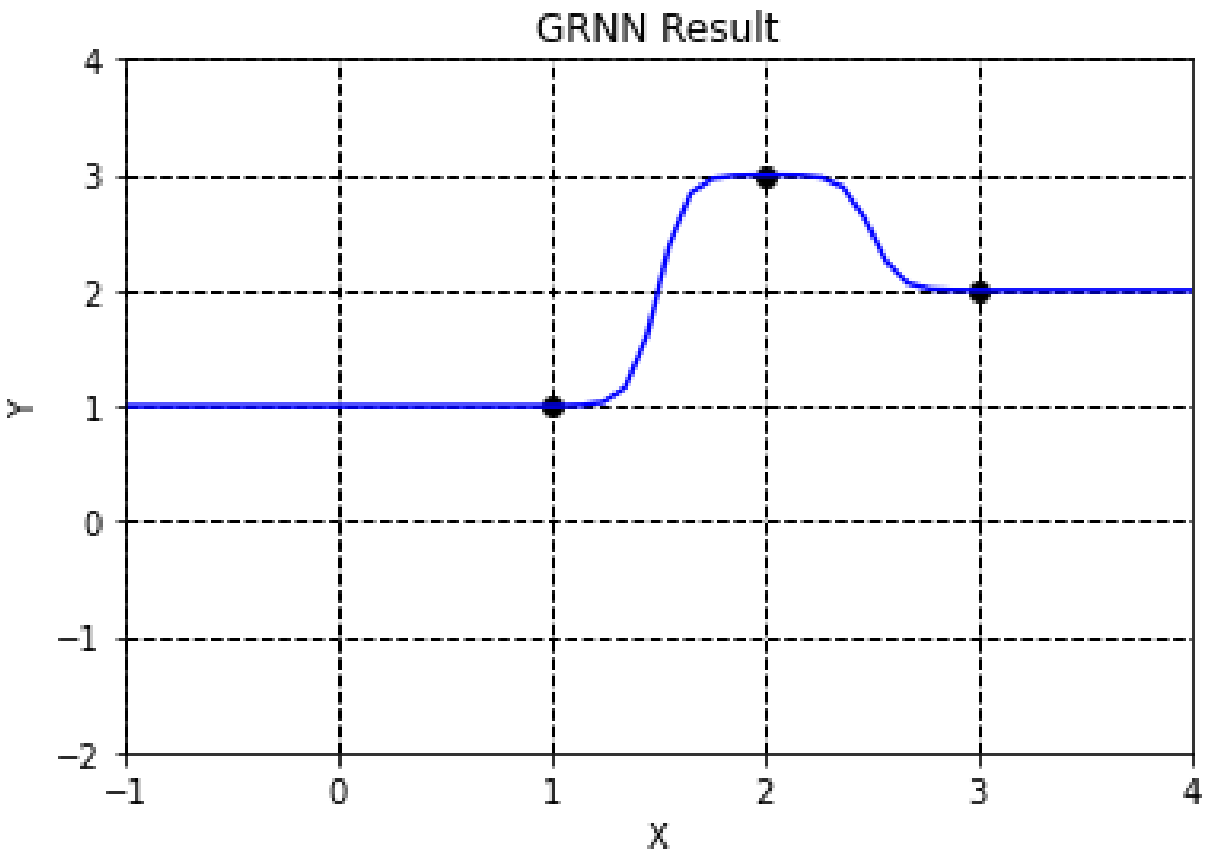
$$\hat{y}(x) = a_1\phi_1 + \ldots + a_N\phi_N(x_i),\ \phi_i = \exp\left[-\frac{(x-x_i)^2}{2\sigma^2}\right], i = 1,\ldots,N,\ \boldsymbol{a} = \left(K + \lambda I_N\right)^{-1}\boldsymbol{y}.$$

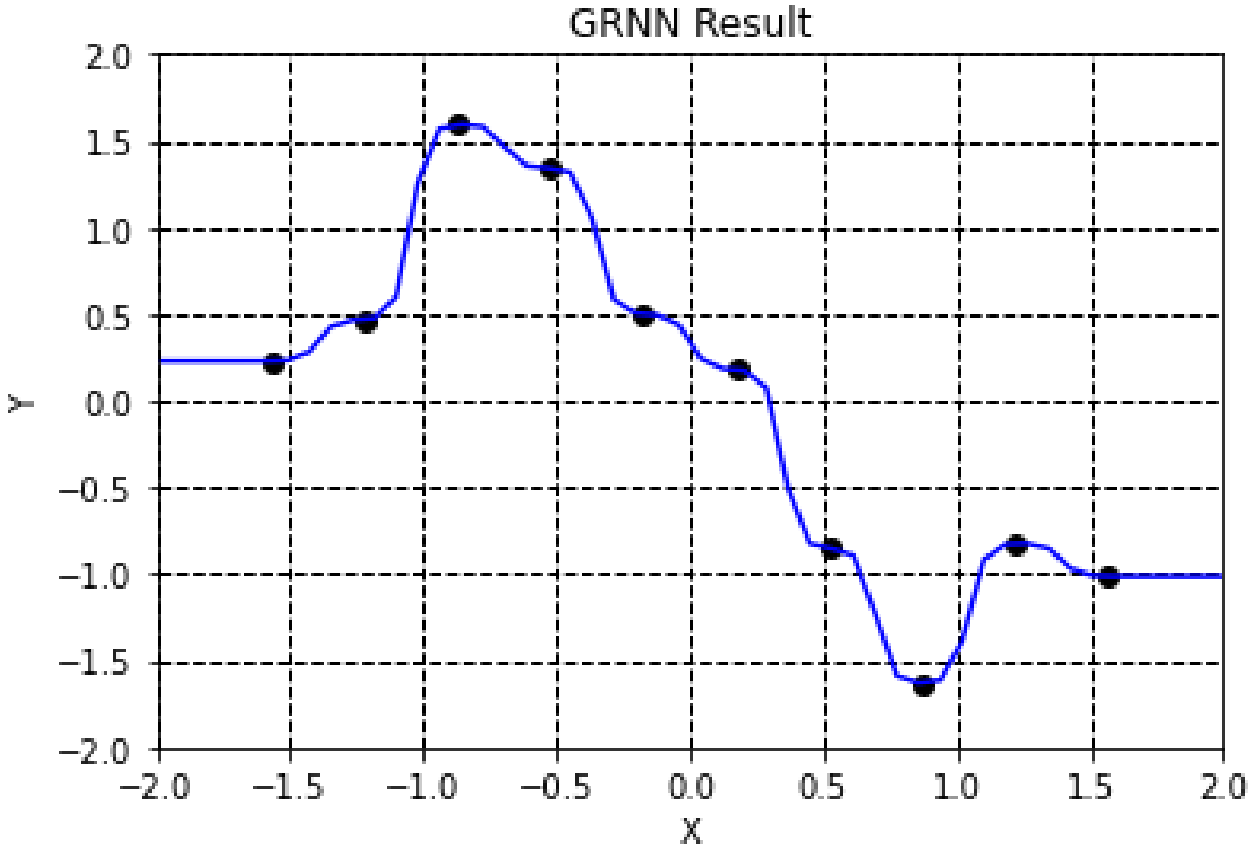
- The GRNN is written as follows:

$$\hat{y}(x) = \frac{y_1\phi_1 + \ldots + y_N\phi_N(x_i)}{\sum_{i=1}^{N}\phi_i},\ \text{where } \phi_i = \exp\left[-\frac{(x-x_i)^2}{2\sigma^2}\right].$$

- In other words, the GRNN is a weighted sum of the input points where the weights themselves are normalized Gaussian kernels.

GRNN result for the three and ten-point examples

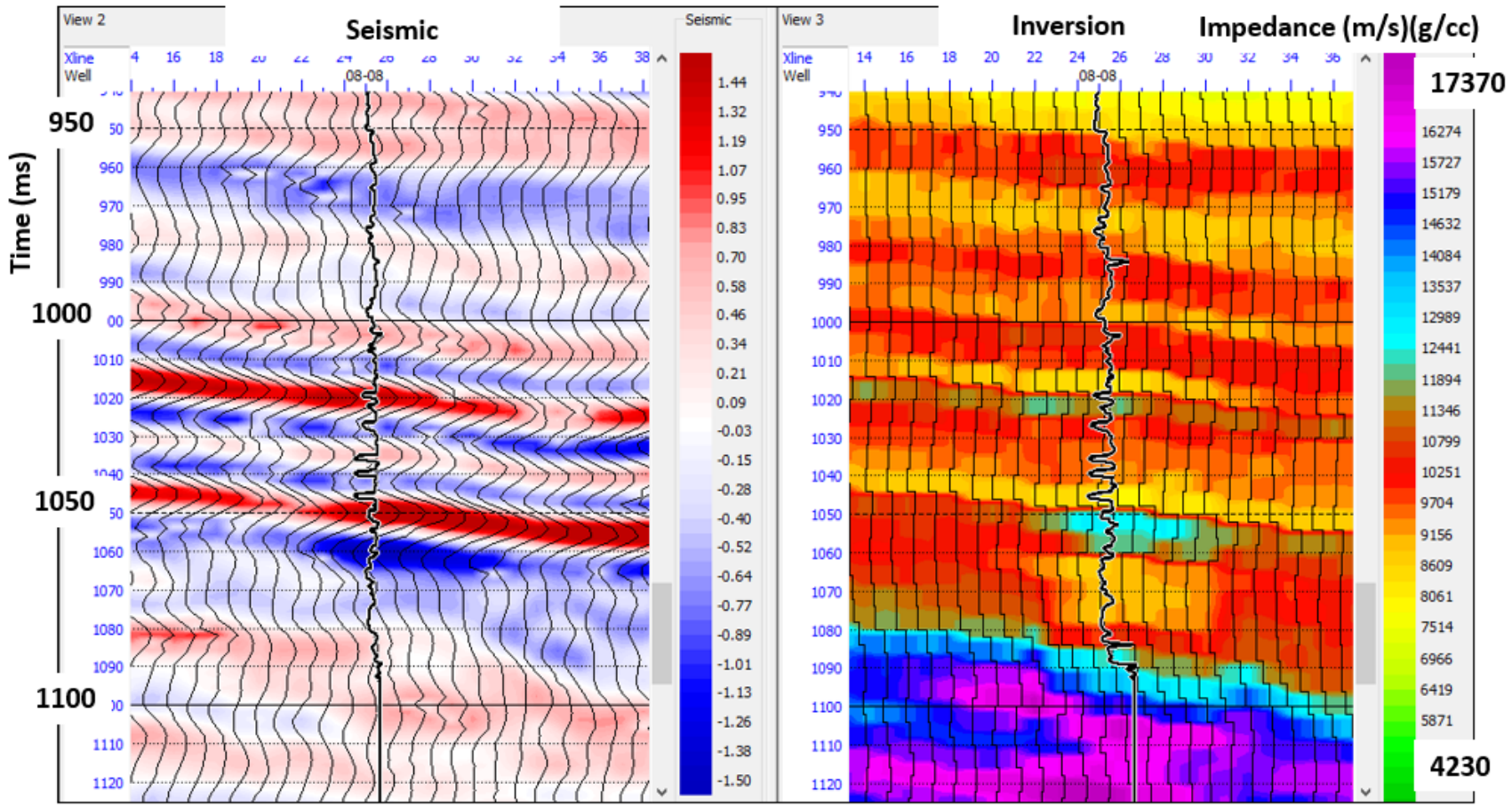




- The application of GRNN for the three and ten-point examples using $\sigma = 0.25$.
- Note that the method extrapolates both the first and last values at the start and end of the plot.

Real data example

- Finally, I will apply several of the techniques we have discussed to the prediction of reservoir parameters from seismic attributes (Hampson et al., 2001).
- I will use a channel sand example from the Western Canadian Sedimentary Basin, in which we predict density logs



A line from the input seismic volume, with the acoustic impedance log spliced in at its location.

A line from the acoustic impedance volume, where the channel sand is located below a time of 1070 msec.

Extending the kernel method to multiple attributes

- The 1D case can be extended to multiple dimensions by letting $\boldsymbol{x}_i$ be the D+1 dimensional row vector for each of the N observed density values, where D is equal to the number of attributes:

$$\boldsymbol{x}_i^T = \begin{bmatrix} 1 & x_{i1} & \dots & x_{iD} \end{bmatrix}, i = 1, \dots, N.$$

- The kernel solution is then given by:

$$\hat{\rho}(\boldsymbol{x}) = a_1\phi_1 + a_2\phi_2 + \dots + a_N\phi_N, \text{ where the two kernels are:}$$

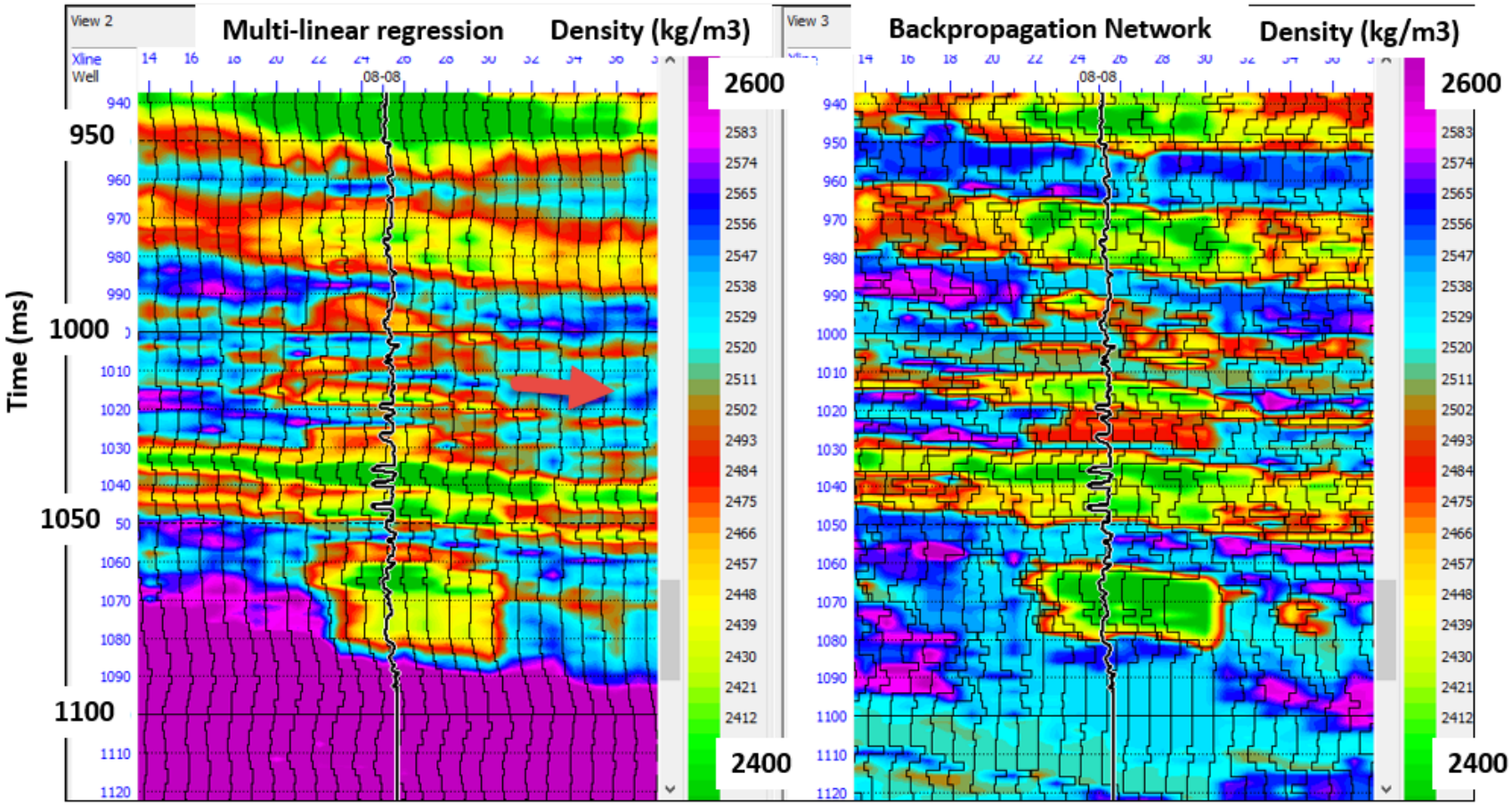
$$\phi_i = (1 + \boldsymbol{x}_i^T \boldsymbol{x}_j)^M, \boldsymbol{a} = \left(K_{ij} + \lambda I_N\right)^{-1} \boldsymbol{\rho}, K_{ij} = (1 + \boldsymbol{x}_i^T \boldsymbol{x}_j)^M, i, j = 1, \dots, N \text{ (Polynomial)}$$

$$\phi_i = \exp\left[-\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{2\sigma^2}\right], \boldsymbol{a} = \left(K_{ij} + \lambda I_N\right)^{-1} \boldsymbol{\rho}, K_{ij} = \exp\left[-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right], i, j = 1, \dots, N \text{ (Gaussian)}$$

- The GRNN approach simply uses the Gaussian kernel: $\phi_i = \exp\left[-\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{2\sigma^2}\right].$

Real data example

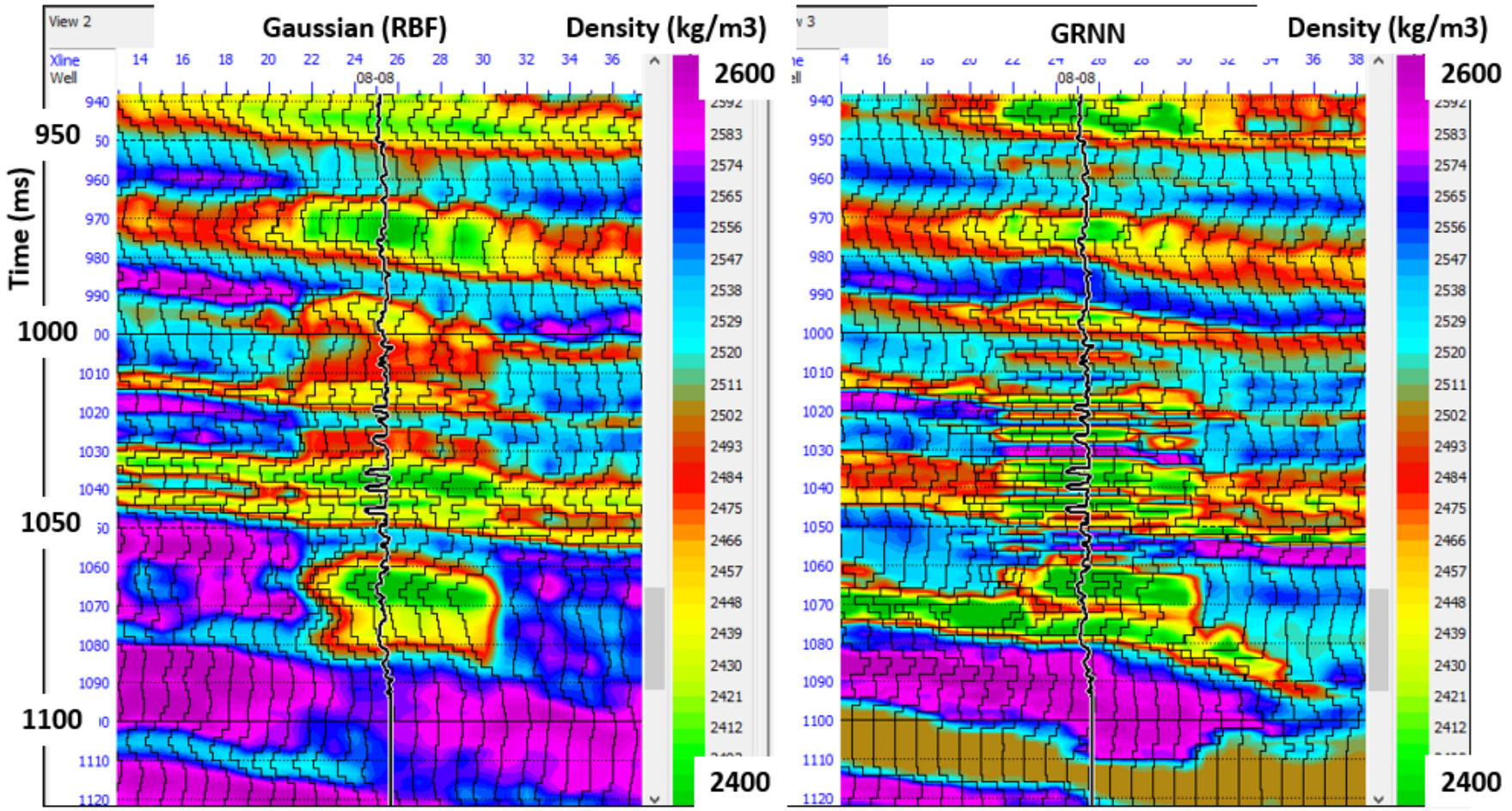
- Here is a comparison between multi-linear regression ($M = 1$) and the backpropagation network, which we did not discuss today.
- The multi-linear regression solution shows more continuity, but the backpropagation network shows higher frequencies.



- I have not yet had a chance to implement the full polynomial kernel method in the real data case but expect it to produce a higher frequency result.

Real data example

- Here is a comparison between the Gaussian kernel regression, or RBF, method and the GRNN approach.
- The GRNN method seems to give the highest frequency result of all four methods, but the RBF method is closer to the previous two results.



- Note that all four analysis methods were able to clearly image the channel sand at a time of 1070 msec.

Conclusions

- In this talk, I discussed the relationship between linear and nonlinear regression techniques.
- I started by introducing the linear basis function model.
- I then compared primal and dual least-squares inversion of this model and showed that the primal approach inverts a much smaller matrix.
- However, by using kernel substitution, I showed that the dual representation leads to two nonlinear regression approaches: polynomial kernel regression and Gaussian kernel regression.
- I then applied these methods to both a simple three-point example and a more complex ten-point noisy sine wave example.
- I then discussed the radial basis function method, which is identical to the Gaussian kernel method, and the related generalized regression method.
- Finally, I applied these approaches to a density prediction problem using a real dataset from Alberta.

Acknowledgements

I wish to thank my colleagues at the CREWES Project and GeoSoftware for their support and ideas, as well as the sponsors of the CREWES Project and NSERC-CRD CRDPJ543578-19 .

References

Bishop, C.M, 2006, Pattern Recognition and Machine Learning: Springer-Verlag.

Hampson, D., Schuelke, J.S., and Quirein, J.A., 2001, Use of multi-attribute transforms to predict log properties from seismic data: Geophysics, 66, 220-231.

Hastie, T., Tibshirani, R., and Friedman, J., 2009, The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd Edition: Springer Series in Statistics.

Murphy. K.P., 2012, Machine Learning: A Probabilistic Perspective: MIT Press, Cambridge, Mass.

Powell, M.J.D., 1987, Radial basis functions for multivariable interpolation: a review. In J.C. Mason and M.G. Cox (Eds.), Algorithms for Approximation, 143-167. Oxford: Clarendon Press.

Ronen, S., Schultz, P.S., Hattori, M., and Corbett, C., 1994, Seismic-guided estimation of log properties, Part 2: Using artificial neural networks for nonlinear attribute calibration: The Leading Edge, 13, Issue 6, 674-678.

Rummelhart, D.E., Hinton, G.E., and Williams, R.J., 1986, Learning representations of back-propagation errors: Nature, 323, 533-536.

Russell, B.H., 2019, Machine learning and geophysical inversion — A numerical study: The Leading Edge, 38, Issue 7, 512-519.

Schölkopf, B. and Smola, A., 2002, Learning with Kernels: MIT Press, Cambridge, Mass.

Shawe-Taylor, J. and Cristianini, N., 2004, Kernel Methods for Pattern Analysis: Cambridge University Press.

Specht, D.F., 1991, A general regression neural network: IEEE Transactions on Neural Networks, 2(6), 568-576.

Appendix: The dual regression solution

- Polynomial regression using the covariance, or inner product, matrix $X^T X$ with ridge regression is called the primal solution and is written:

$$\boldsymbol{w} = \left(X^T X + \lambda I_{M+1} \right)^{-1} X^T \boldsymbol{y}$$

- To derive the dual solution, we start by bringing the inverted term back to the left side:

$$\left(X^T X + \lambda I_{p+1} \right) \boldsymbol{w} = X^T \boldsymbol{y}$$

- This can then be re-arranged as: $X^T X \boldsymbol{w} + \lambda \boldsymbol{w} = X^T \boldsymbol{y}$

- Further manipulation gives: $\boldsymbol{w} = \lambda^{-1} \left(X^T \boldsymbol{y} - X^T X \boldsymbol{w} \right)$
- Factoring out the transpose of X then gives

$$\boldsymbol{w} = X^T \lambda^{-1} \left(\boldsymbol{y} - X \boldsymbol{w} \right)$$

Appendix: The dual regression solution

- This can be written as: $\boldsymbol{w} = X^T \boldsymbol{a}, \text{ where } \boldsymbol{a} = \lambda^{-1}\left(\boldsymbol{y} - X\boldsymbol{w}\right)$
- Now comes the "trick", where we re-substitute the $X^T\boldsymbol{a}$ form into $\boldsymbol{w}$:

$$\lambda\boldsymbol{a} = \boldsymbol{y} - X\boldsymbol{w} = \boldsymbol{y} - XX^T\boldsymbol{a}$$

- This can then be re-arranged as: $\lambda\boldsymbol{a} = \boldsymbol{y} - XX^T\boldsymbol{a}$
- We then group the $\boldsymbol{a}$ terms: $\left(XX^T + \lambda\right)\boldsymbol{a} = \boldsymbol{y}$
- We now re-substitute this into the definition of $\boldsymbol{w}$ to get:

$$\boldsymbol{a} = \left(XX^T + \lambda I_N\right)^{-1}\boldsymbol{y}$$

- Finally, re-substitute into the weight equation to get the final form of the dual regression solution:

$$\boldsymbol{w} = X^T\boldsymbol{a} = X^T\left(XX^T + \lambda I_3\right)^{-1}\boldsymbol{y}$$

Appendix: The dual regression solution

- That was a complicated derivation, but all you need to remember is the final equation for dual regression:

$$\boldsymbol{w} = X^T \left(XX^T + \lambda I_N \right)^{-1} \boldsymbol{y}$$

- Compare this with the equation for primal regression:

$$\boldsymbol{w} = \left(X^T X + \lambda I_{p+1} \right)^{-1} X^T \boldsymbol{y}$$

- Notice that this tells us that:

$$X^T \left(XX^T + \lambda I_N \right)^{-1} = \left(X^T X + \lambda I_{p+1} \right)^{-1} X^T$$

- That is, the weights can be computed using either an inverse containing the *N* x *N* matrix XX^T or the *p*+1 x *p*+1 matrix $X^T X$.