# Project Description

Scoliosis is quantified in terms of the Cobb-angle. This is the maximum angle between the end plates of any two vertebrae. As outlined in our project proposal, our goal is to produce a method for estimating the curvature of scoliotic spines using ultrasound landmark data with neural networks. The ultrasound landmark data consists of sets of 2n 3D points, where n is the number of vertebrae included in that patient’s scan. 2n is the number of transverse process landmarks located from the scan.

As a prototype of such a method, we developed a pre-processing procedure which makes any such landmark set suitable for use with a layered, feedforward neural network. The pre-processing procedure consists first of centering the landmarks sets, creating a set with points for vertebrae that could be present with any patient, creating a set with simulated ultrasound error, splicing the new sets together, and normalizing the resulting landmark set. These steps are illustrated and explained in greater detail in Appendix 1 - Data Pipeline. The entire pre-processing component is implemented as a 3DSlicer (<https://www.slicer.org/>) extension for the environment’s batch processing and visualization capabilities.

A feedforward neural network was chosen for the angle estimation component of the pipeline prototype for its function approximation capabilities and the functionality offered by MATLAB for assessment of the prototype. By creating landmark sets complete with points corresponding to those that could be found in any patient’s scan (the transverse processes of Thoracic 1 to Lumbar 5), constant input data dimensionality is achieved, and this network architecture can be used. The neural network component is illustrated and explained in greater detail in Appendix 1 - Data Pipeline.

To evaluate the performance of our prototypical method, we generated several data sets, trained the network on them, and observed the training, validation, and in particular, testing mean-squared-errors (MSE). The control set consisted of only the normalized, completed, centered sets. Random noise of increasing standard deviation was added to the original landmark locations to create the remaining sets. The network was trained and retrained multiple times with each data set to collect MSE values. Plots of these results are shown in Appendix 2 - Results.

This is a step towards demonstrating the robustness of such a method. The method must accurately estimate the true curvature even with imperfect data, as is likely with ultrasound. Our results show that the mean testing MSE for the network remains within the clinically acceptable limit of error of 5o for the tested range of [0, 10] mm2 noise standard deviations.

Work to improve the robustness, and extent of validation, of the method is ongoing. A module for estimating missing point locations (sometimes a problem with ultrasound) using either statistical, or another neural method, is planned for implementation between the ultrasound error simulation, and set splicing, as they are shown in Figure 2. A means of partitioning the patients’ landmark sets into testing and training sets sorted by severity of curvature is also planned. This will provide valuable information regarding the relationship between the method’s performance and severity of the disease. This will be useful information since other work has focused on cases with curvatures typically less than 45o, whereas we have a number of cases worse than this. We do not expect our final neural network to be implemented using MATLAB’s app. Work has begun on a Visual Studio C++ solution which will implement our network. This will provide more freedom to experiment with different network architectures and learning strategies.

Ben programmed the pre-processing modules as he is familiar with Python development in 3DSlicer. Ben also handled the result generation using MATLAB. Kyle worked on the inclusion of appropriate methods for dealing with some of the sources of data error (eg. missing data values).

Appendix 1 - Data Pipeline

For clarity, the data pipeline is shown from the highest level in Figure 1. The pre-processing is expanded in Figure 2 and the result of preprocessing is illustrated in Figure 3. The neural network is expanded in Figure 4, and shown with landmark points in Figure 5.



Figure 1: Top-level view of data analysis pipeline

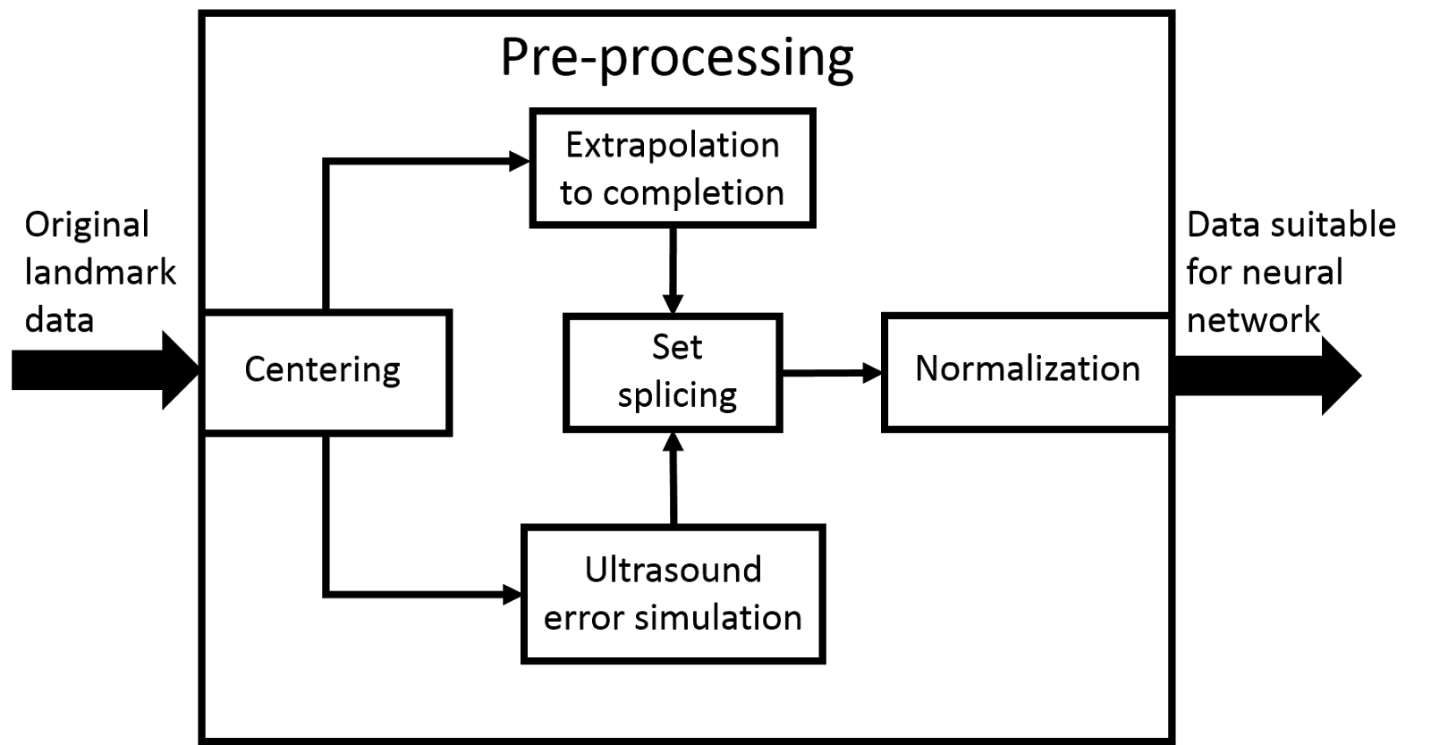


Figure 2: Modules used in pre-processing component of procedure

The centering program is written in ScoliosisNeuralNets/CenterAndNormalizeLandmarks/CenterAndNormalizeLandmarks.py. It simply finds the mean coordinate for each dimension of each patient’s landmarks, and subtracts that value from that coordinate of all landmarks.

Ultrasound error simulation was done with the Slicer module in ./ScoliosisNeuralNets/DegradeTransverseProcesses/DegradeTransverseProcesses.py. Currently, the only error introduced is random noise is each coordinate of each landmark. We plan to introduce another pre-processing step which is capable of estimating and imputing missing values, and correcting misplaced points if necessary.

The partially complete, centered original landmarks must be extrapolated to completion if a neural network with constant architecture is to be used. This is done in the Slicer module in ./ScoliosisNeuralNets/ExtendSpine/ExtendSpine.py. It works by taking the top-most, and bottom-most pair of landmarks, and placing points with the same relative position at multiples of the vector leading from the second outermost landmark points to outermost landmark points, until the set is complete. This ensures that the extrapolation does not affect the curvature encoded in the landmarks, since they add no new angles, and a minimum of features.

If we are to deal with missing or misplaced values in the future, they must be repaired by this point. The landmarks of the completed set which have corresponding points in the set with simulated error are replaced by those with simulated error. Errors were not introduced into the extrapolated points as they were meant for place holders, not to affect results. This replacement of points, called “Splicing” in Figure 2, is performed in the Slicer module in ./ScoliosisNeuralNets/ExtendSpine/ExtendSpine.py.

Finally, each patient’s landmark points are normalized by the Slicer module in ./ScoliosisNeuralNets/CenterAndNormalizeLandmarks/CenterAndNormalizeLandmarks.py. This is simply a matter of finding the maximum absolute coordinate value in each dimension, and dividing all landmark coordinates in that dimension, for that patient, by that value. A normalized (and completed) landmark set is shown on the right of Figure 3.



Figure 3: Original landmarks with CT derived model to demonstrate corresponding spinal anatomy, and centered (undiscernible), completed, and normalized (almost undiscernible) landmark set resulting from pre-processing.

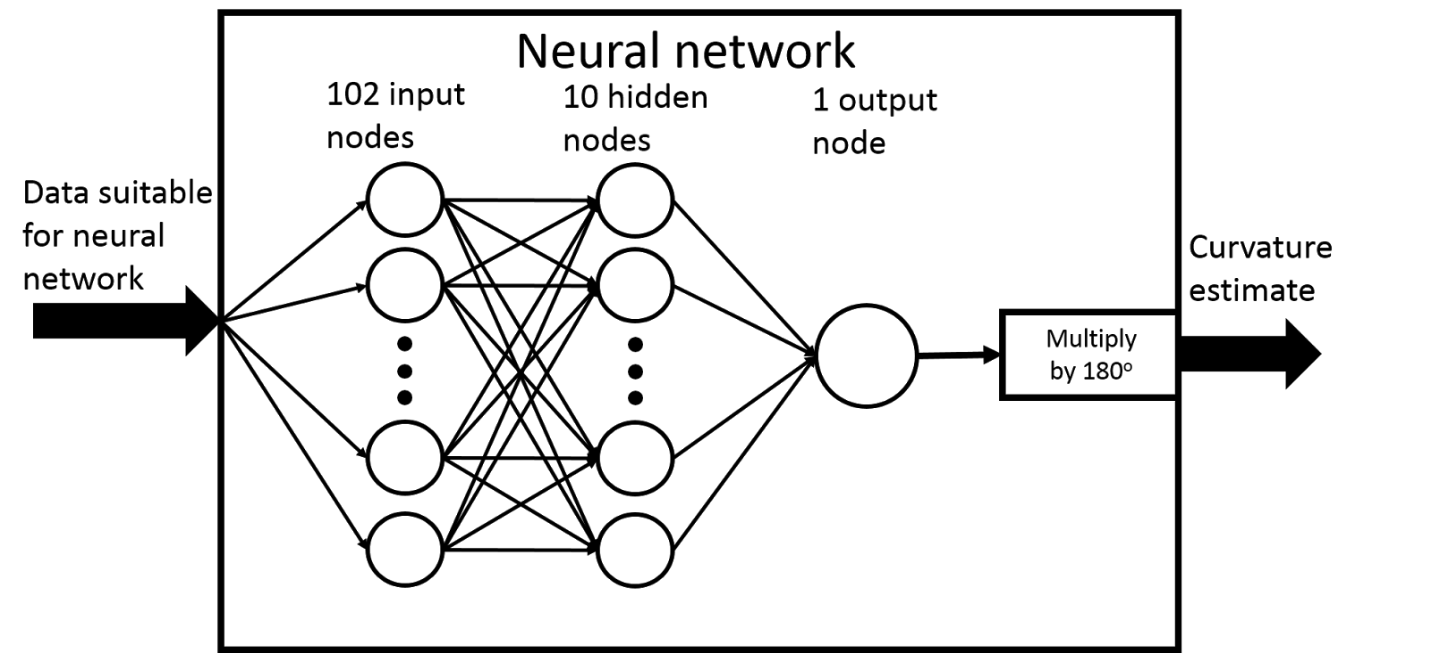


Figure 4: Feedforward neural network architecture

The expansion of the neural network in Figure 4 shows its architecture. Each normalized coordinate of each landmark was used as input to one input node. These landmarks are shown in Figure 5. Specifically, 102 values in the range [-1, 1] were used as input. 102, being the number of landmarks in a complete set, 34, times the 3 spatial dimensions of each. One output node was needed to obtain an angle estimate. The output node’s activation was multiplied by 180o, mapping its [0, 1] output to the range [0o, 180o]. The weight values were trained using MATLAB’s Lavenberg-Marquardt training algorithm with 10 hidden nodes. 15% of the landmark sets were used for MATLAB’s validation, and 15% for testing. With 124 patients’ landmarks, that corresponds to 19 sets for validation and testing, and 86 for training. These are simply MATLAB’s default parameters and were used as a starting point for result generation.



Figure 5: Illustration of 3D point values to be used as input to produce curvature estimate output, 34 in total, 2 on each vertebra.

# Appendix 2 - Results

Figure 6: Mean-squared-error in angle output, averaged over 10 training trials, for a range of noise standard deviations

Figure 7: Standard deviation, averaged across 10 training trials, of the testing MSEs, for a range of noise standard deviations

The result of MATLAB’s Neural Network Function Fitting app was the mean-squared-error of the training, validation, and test sets. The test set errors, being the results of interest, were averaged over 10 training trials for each experimental configuration. 11 experimental configurations were used, one for each additional mm2 in the points’ noise’s standard deviation from 0 to 10. Figure 6 shows a modest increase in the MSE with noise. The resulting average error is soundly less than the clinically acceptable limit of error, 5o, as discussed in our project proposal.

However, the results ought not to be compared directly. Our method recovers the angle extracted from ideal ultrasound data, whereas the Cobb angle can only be observed directly with X-ray. Authors mentioned in our proposal validated their results against X-ray ground-truth. We do not have access to X-ray data, just the landmark points. Therefore, we cannot remark on the superiority of either method. Fortunately, the intention was not to replace existing methods, but to supplement them with a tool for dealing with imperfect/incomplete data. Our method demonstrates robustness, so far, against noise, and stands soon to be tested against other challenges.