**Multi-Objective Optimization of Iron Ore Induration process using optimal neural networks**

## 1. Sobol Sampling plan and Criteria for uniformity

Sobol sampling originates from quasi-random sequence generator, designed to create a mesh of uniformly distributed N sample points for solving Eq S1 by approximating it with Eq S2.

(S1)

(S2)

is a point in k dimensional unit cube . For better approximation of integrands in Eq S1 having a wide range in , needs to discretized into uniform mesh. Among several criteria of uniformity, discrepancy was considered. The counting function is defined as the number of points in . For each point , a k-dimensional rectangle is constructed with each side formed by the segment . Discrepancy is then defined as

(S3)

Lower the discrepancy, higher is the uniformity. The steps to generate 1 dimensional Sobol sequence are described next.

*Step 1. Generation of Directional Numbers*

Define a set of real numbers , called the directional numbers using the coefficients in a primitive polynomial of degree d as shown in Eqs S4 to S6. The coefficients in Eq S5 can either be 0 or 1 with and as an odd positive integer < 2i while is bitwise exclusive-or operator.

(S4)

(S5)

(S6)

*Step 2. Generation of Sobol sequence*

Once the directional numbers are generated, the sequence of sample points can be obtained by Eq S7 where **j** in binary code = b1b2b3…

(S7)

*Remark 1.* This method can be generalized to k dimensions by generating k sequences using different primitive polynomials leading to different directional vectors.

*Remark 2.* By Gray coding the directional numbers instead of binary coding, it can be shown that,

(S8)

*Remark 3.* This procedure provides a unique property to Sobol sampling plan. Unless the primitive polynomial in Eq S4 is changed, the sequence of samples generated in Eq S7 remains same, irrespective of the size of the samples.

## 2. Sample size determination (SSD) algorithms

Start the following algorithms with the values of constants α, β and γ as 10, 0.3 and 3, respectively. k is the number of input dimensions.

## 2.1 Algorithm 1 – Hypercube Sampling based SSD.

*Initialization.* Set N = αk | N ∈ (S9)

*Step 1. Sobol sequence*

Obtain the set S of N points from Sobol distribution.

(S10)

Project S on using normalizing function such that each entity in is normalized between UB and LB (k-dimensional vectors of upper and lower bounds of inputs, respectively).

(S11)

Define to be the set of output points obtained from

(S12)

(S13)

*Step 2. Hypercube formation*

Construct a hypercube defined as,

(S14)

This ensures where is the region bounded by .

*Step 3. Sampling*

Divide the hypercube into hyper cubes ; that is,

(S15)

Define two null sets and . Identify the set of points in that are contained in each . Randomly select one point from the region bounded by each , find its corresponding point in set and add it to . Then,

(S16)

*Step 4. Tolerance Criteria*

The given ANN is trained by set and validated with ,

(S17)

(S18)

Then define the error metric,

(S19)

Evaluate the metric , if current iterate value 2.

(S20)

are Error metrics of current and previous iteration, Size of current and previous iteration, respectively.

*Step 5. Iteration or Termination*

If tolerance, terminate the algorithm, report the sample size to be N. Else, set N = N+ and go to Step 1.

## 2.2 Algorithm 2 – Single objective optimization problem (SOOP) based SSD.

The Initialization and Step 1 are same as those of Algorithm 1. Steps 3-4 are same as Steps 4-5 of Algorithm 1, respectively. Novelty in Step 2 is as follows.

*Step 2. Optimization formulation*

Given the set of size N, select points from it such that the min-max metric [19] (defined in manuscript) is minimized; that is,

(S21)

The remaining input points constitute the set , as in Eq S16.

## 2.3 Algorithm 3.

The Initialization and Steps 1-3 are same as that of Algorithm 1. The novelty in Step 4 is as follow.

*Step 4. Solving the SOOP*

Call the set from Step 3 as and solve Step 2 of Algorithm 2 by giving as the initial guess to the optimizer.

Steps 5-6 are same as Steps 4-5 of Algorithm 1, respectively.

## 2.4 Algorithm 4 – Validation Set based SSD.

Given the MLP architecture, this algorithm finds the optimal sample size of , sampled using Sobol scheme, such that when trained by , ANN has minimum validation error with Held-out set .

(S22)

*Remark 4.* One common aspect of all sample size determination algorithms discussed so far is the ever changing N across the iterations. If a sampling plan such as LHS is used, total function calls will be cumulative sum of sample sizes at each iteration, because a new sequence is generated every time when N changes. However, with Sobol, due to the reason mentioned in *Remark 3*, the sequence of sample points doesn’t change with changes in N resulting in no wastage of sample points of previous iterations, thus reducing the precious function calls.

## 3. Comparison of Sobol sample plan with LHS sampling plan

Figure S1 presents the results of comparison between Sobol and Latin Hypercube Sampling (LHS) [19] in terms of computational time and metric (defined in manuscript) [19]. Quasi random Monte Carlo Simulations to the order of 2.5e5 were conducted to approximate the true mean of Throughput F using Eq S23 to 401.6. and are the Lower and Upper bounds of inputs

(S23)

Figure S2 depicts the convergence of expectations evaluated by Sobol, LHS and Random sampling plans to the true mean. These results conclude that Sobol has better computational advantage, more uniformity and faster rate of convergence than LHS. Further, as reported in Remark 4, Sobol ensures no wastage of sample points during iterative SSD algorithms. Sobol was therefore used in our proposed algorithms instead of conventional Design of Experiments (DoEs).

## 4. Comparison between Sobol based sample size determination algorithms

Figure S3 presents a comparison between proposed sample size determination algorithms for a fixed ANN architecture**.** It is observed that Algorithm 1 is the fastest among all proposed SSD algorithms. However, being random sampling based technique it can be considered inferior to Algorithm 2 in terms of theoretical basis due to the involvement of optimization formulation in Algorithm 2. As shown in Figure S3, computational speed advantage of Algorithm 1 along with theoretical basis of Algorithm 2 makes Algorithm 3 an efficient sample size determination technique.

## 5. Tests for overfitting

In each of the following tests, the architectures which came up as candidates for evaluation in generations 1-5 of NSGA II algorithm for solving optimal design problem of ANN (Eq 1 in manuscript) were compared with the resultant optimal ANNs in Tables S1 to S3. We next describe each of these tests in details.

## 5.1 Information theory test.

Akaike and Bayesian Information Criteria (AIC and BIC) are used to check the credibility of ANN architectures [20]. The AIC and BIC measures are established techniques in information theory analysis known to identify overfitted models.

(S24)

(S25)

These criteria penalize the model for containing large number of parameters, P. In case of MLPs, P = number of weights and biases, S is Sample size for training and MSE is training Mean Square Error. Lower the value of these measures, less overfitted is the model.

## 5.2 Cross validation test.

This versatile technique starts with dividing the given dataset into K folds or groups containing equal number of sample points. MLP model is then trained by K-1 folds and validated with data in left over fold. Since K-1 folds can be selected in K different ways, the Cross Validation (CV) error is evaluated by taking the mean of K validation errors. The extensive training and validation of ANN in all the regions of input domain ensures thorough testing of the model. If the model is overfitted, CV error will be higher.

## 5.3 Held-Out Sample test.

In this method, trained MLP model is tested against a set of data completely different from the dataset used for training. A set of 100 points were randomly sampled to constitute the held-out set.

## 5.4 Test with noisy data.

Given the sample size by SSD algorithm, the MLPs were retrained after introducing white noise in training output, such that the Signal to Noise Ratio (SNR) is maintained at 20.25 decibels.

(S26)

The results are presented in Figure S4 of the manuscript. For the noisy data, the Predicted R2 value for Algorithm 4, 3 and 1 was found to be 0.9972, 0.9912 and 0.9891, respectively. Lower SNR values were not considered, because it is not claimed that proposed algorithms filter noise and thus become capable of emulating noisy data. Instead, noisy data was only used as a test for checking overfitting.

## 5.5 Goodness of Fit test.

Regular and Predicted are defined in Eq S27, with only difference in the data set used for their evaluation. Regular is the correlation coefficient with respect to the training data. This measure improves continuously with increase in training data, making it unfit for checking over-fitting. Predicted , on the other hand, is with respect to test set and thus, can be used in Held-Out sample test along with RMSE. Apart from these, Adjusted is also defined in Eq. S28 which unlike the previous two, penalizes the model for increase in parameters, similar to the philosophy of AIC and BIC measures defined in Eqs S24 and S25, respectively.

(S27)

.

are MLP model, Physics based model and Held-out dataset 200 LHS points, respectively.

(S28)

Adjusted value increases with increase in sample size (N) and/or parameters (P) till the optima is found. After that, the Adjusted decreases, indicating overfitting. Among the selected models, the one having maximum value of Adjusted can be considered as the least overfitted model.

Finally, the results of emulation with optimal MLPs and novel SSD methods (reported in Table 1 of manuscript) are compared with the predictability of same architectures when trained by specific techniques known in literature to prevent overfitting such as Early stopping and Bayesian Regularization. These results are presented in Table S4.

## Figures and Tables

D:\induration_paper\Figures\fig2.tif

Figure S1. Comparison of Sobol and LHS sampling plan. a) Computational time comparison b) Min-Max metric comparison.

D:\induration_paper\Figures\SP_output1.tif

Figure S2. Convergence of Expectation of Sobol, LHS and Random sequence to true mean value obtained by Monte Carlo Simulations.

D:\induration_paper\SSD_comparison.tif

Figure S3. Comparison of SSD Algorithms in terms of computational speed. Algorithm 1 to 3 were run for arbitrarily selected architecture [22-3-4-6-1] with tan sigmoidal activation for emulation of Throughput. Although each algorithm determined a marginally different value, they were restricted to a fixed sample size, for an unbiased comparison. It can be observed that Algorithm 2 is slowest. However, an intelligent initial guess, such as the solution of Algorithm 1 –, makes it more efficient. This implementation in Algorithm 3 gives it immense computational advantage compared to Algorithm 2.

D:\induration_paper\Noise_test_20_nov.tif

Figure S4. Test with Noisy data. Training and Validation parity plots for emulating Throughput using ANN with a) Algorithm 1, b) Algorithm 3 and c) Algorithm 4. Even after training the ANNs with noisy data having Signal to Noise Ratio (SNR) = 20.25 Decibels, the Held Out set validation RMSE was found to be less than 5% for all the SSD algorithms. The Predicted R2 value for Algorithm 4, 3, 1 was found to be 0.9972, 0.9912 and 0.9891, respectively.

Table S1. Information Theory tests for overfitting performed with respect to emulation of throughput. ANN is the architecture being tested. is the transfer function choice, P is number of parameters and N is estimated sample size. MSE is training mean square error. The ANN architectures in bold are optimal ANNs which are observed to outperform other architectures.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ANN |  | P | N | Information Theory Test | | |
| MSE | AIC | BIC |
| Algorithm 1 | | | | | | |
| 22-1-1-6-1 | 1 | 65 | 370 | 1.5E-4 | -3107.5 | -2853.2 |
| 22-2-0-0-1 | 1 | 68 | 240 | 4.1E-5 | -2288.3 | -2051.6 |
| 22-2-1-2-1 | 1 | 77 | 590 | 0.0017 | -3608.5 | -3271.2 |
| **22-3-5-2-1** | **2** | **125** | **240** | 2.9E-8 | **-3911.8** | **-3476.7** |
| 22-7-4-1-1 | 1 | 221 | 430 | 4.4E-5 | -3863.7 | -2965.6 |
| 22-7-6-1-1 | 1 | 239 | 350 | 0.014 | -1012.9 | -90.9 |
| 22-6-6-6-1 | 2 | 250 | 340 | 9.9E-6 | -3414.7 | -2457.5 |
| Algorithm 3 | | | | | | |
| 22-3-5-5-1 | 2 | 146 | 250 | 9.9E-7 | -3162.6 | -2648.4 |
| 22-6-1-3-1 | 1 | 176 | 260 | 2.7E-5 | -2379.4 | -1752.8 |
| **22-6-4-0-1** | **2** | **188** | **280** | **9.6E-9** | **-4791.4** | **-4108.1** |
| 22-6-4-2-1 | 1 | 200 | 440 | 5.4E-5 | -3917.6 | -3100.3 |
| 22-8-3-0-1 | 2 | 233 | 300 | 9.9E-7 | -3679.4 | -2816.4 |
| 22-8-4-7-1 | 1 | 284 | 640 | 6.3E-5 | -5622.6 | -4355.5 |
| 22-8-5-7-1 | 1 | 300 | 810 | 4.6E-5 | -7485.0 | -6075.8 |
| Algorithm 4 | | | | | | |
| **22-3-0-0-1** | **2** | **91** | **1056** | **7.3E-7** | **-14729** | **-14278** |
| 22-2-4-6-1 | 2 | 116 | 1056 | 4.1E-5 | -10457 | -9881.5 |
| 22-3-3-3-1 | 2 | 118 | 1056 | 3.8E-5 | -10492 | -9906.7 |
| 22-3-5-6-1 | 2 | 153 | 1056 | 3.7E-5 | -10421 | -9661.9 |
| 22-5-6-3-1 | 1 | 197 | 1066 | 8.6E-5 | -9585.5 | -8606.1 |
| 22-5-7-5-1 | 1 | 224 | 1193 | 1.1E-4 | -10467 | -9327.7 |
| 22-8-3-6-1 | 1 | 263 | 1706 | 1.3E-4 | -14675 | -13244 |

Table S2. K fold Cross Validation Tests for overfitting performed with respect to emulation of throughput. ANN is the architecture being tested. is the transfer function choice, P is number of parameters and N is estimated sample size, K is number of folds. The ANN architectures in bold are optimal ANNs which are observed to outperform other architectures.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ANN |  | P | N | K Fold CV Test | | |
| K=10 | K = 5 | K = 3 |
| Algorithm 1 | | | | | | |
| 22-1-1-6-1 | 1 | 65 | 370 | 0.179 | 0.182 | 0.143 |
| 22-2-0-0-1 | 1 | 68 | 240 | 0.187 | 0.123 | 0.061 |
| 22-2-1-2-1 | 1 | 77 | 590 | 0.236 | 0.181 | 0.124 |
| **22-3-5-2-1** | **2** | **125** | **240** | **0.084** | **0.059** | **0.033** |
| 22-7-4-1-1 | 1 | 221 | 430 | 0.197 | 0.152 | 0.106 |
| 22-7-6-1-1 | 1 | 239 | 350 | 0.156 | 0.142 | 0.173 |
| 22-6-6-6-1 | 2 | 250 | 340 | 0.097 | 0.062 | 0.057 |
| Algorithm 3 | | | | | | |
| 22-3-5-5-1 | 2 | 146 | 250 | 0.109 | 0.093 | 0.083 |
| 22-6-1-3-1 | 1 | 176 | 260 | 0.189 | 0.132 | 0.116 |
| **22-6-4-0-1** | **2** | **188** | **280** | **0.109** | **0.096** | **0.067** |
| 22-6-4-2-1 | 1 | 200 | 440 | 0.210 | 0.199 | 0.162 |
| 22-8-3-0-1 | 2 | 233 | 300 | 0.110 | 0.095 | 0.073 |
| 22-8-4-7-1 | 1 | 284 | 640 | 0.196 | 0.156 | 0.122 |
| 22-8-5-7-1 | 1 | 300 | 810 | 0.231 | 0.196 | 0.128 |
| Algorithm 4 | | | | | | |
| **22-3-0-0-1** | **2** | **91** | **1056** | **0.067** | **0.054** | **0.044** |
| 22-2-4-6-1 | 2 | 116 | 1056 | 0.087 | 0.054 | 0.049 |
| 22-3-3-3-1 | 2 | 118 | 1056 | 0.098 | 0.053 | 0.050 |
| 22-3-5-6-1 | 2 | 153 | 1056 | 0.069 | 0.055 | 0.058 |
| 22-5-6-3-1 | 1 | 197 | 1066 | 0.089 | 0.076 | 0.067 |
| 22-5-7-5-1 | 1 | 224 | 1193 | 0.097 | 0.083 | 0.068 |
| 22-8-3-6-1 | 1 | 263 | 1706 | 0.076 | 0.043 | 0.040 |

Table S3. Hel-out sample tests for overfitting performed with respect to emulation of throughput. ANN is the architecture being tested. is the transfer function choice, P is number of parameters and N is estimated sample size. RMSE is root mean square error for test set. The ANN architectures in bold are optimal ANNs which are observed to outperform other architectures.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ANN |  | P | N | Held Out Sample test | | | |
| RMSE |  |  |  |
| Algorithm 1 | | | | | | | |
| 22-1-1-6-1 | 1 | 65 | 370 | 0.1402 | 0.99901 | 0.99871 | 0.9988 |
| 22-2-0-0-1 | 1 | 68 | 240 | 0.1391 | 0.99931 | 0.99901 | 0.9987 |
| 22-2-1-2-1 | 1 | 77 | 590 | 0.1341 | 0.99982 | 0.99976 | 0.9998 |
| **22-3-5-2-1** | **2** | **125** | **240** | **0.0612** | **0.99991** | **0.99981** | **0.9998** |
| 22-7-4-1-1 | 1 | 221 | 430 | 0.1460 | 0.99991 | 0.99981 | 0.9981 |
| 22-7-6-1-1 | 1 | 239 | 350 | 0.1348 | 0.99991 | 0.99971 | 0.9978 |
| 22-6-6-6-1 | 2 | 250 | 340 | 0.0751 | 0.99992 | 0.99969 | 0.9998 |
| Algorithm 3 | | | | | | | |
| 22-3-5-5-1 | 2 | 146 | 250 | 0.0691 | 0.9990 | 0.99758 | 0.9987 |
| 22-6-1-3-1 | 1 | 176 | 260 | 0.1398 | 0.9996 | 0.99875 | 0.9970 |
| **22-6-4-0-1** | **2** | **188** | **280** | **0.0663** | **0.9999** | **0.99969** | **0.9998** |
| 22-6-4-2-1 | 1 | 200 | 440 | 0.1580 | 0.9999 | 0.99981 | 0.9815 |
| 22-8-3-0-1 | 2 | 233 | 300 | 0.0664 | 0.9999 | 0.99954 | 0.9998 |
| 22-8-4-7-1 | 1 | 284 | 640 | 0.1565 | 0.9999 | 0.99982 | 0.9614 |
| 22-8-5-7-1 | 1 | 300 | 810 | 0.1547 | 0.9999 | 0.99984 | 0.9833 |
| Algorithm 4 | | | | | | | |
| **22-3-0-0-1** | **2** | **91** | **1056** | **0.0293** | **0.9999** | **0.99989** | **0.9999** |
| 22-2-4-6-1 | 2 | 116 | 1056 | 0.0449 | 0.9999 | 0.99988 | 0.9987 |
| 22-3-3-3-1 | 2 | 118 | 1056 | 0.0450 | 0.9999 | 0.99988 | 0.9987 |
| 22-3-5-6-1 | 2 | 153 | 1056 | 0.0451 | 0.9999 | 0.99988 | 0.9987 |
| 22-5-6-3-1 | 1 | 197 | 1066 | 0.0305 | 0.9999 | 0.99987 | 0.9983 |
| 22-5-7-5-1 | 1 | 224 | 1193 | 0.0355 | 0.9999 | 0.99987 | 0.9962 |
| 22-8-3-6-1 | 1 | 263 | 1706 | 0.0443 | 0.9999 | 0.99988 | 0.9984 |

Table S4. Tests for overfitting performed with respect to emulation of throughput and comparison with existing training methods in Neural Network Toolbox

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Training  Method | Algorithm 1  [22-3-5-2-1] = 2  Parameters = 125  Sample Size = 240 | | | Algorithm 3  [22-6-4-0-1] = 2  Parameters = 188  Sample Size = 280 | | | Algorithm 4  [22-3-0-0-1] = 2  Parameters = 91  Sample Size = 1056 | | |
| Training MSE | AIC | Test Set RMSE | Training MSE | AIC | Test Set RMSE | Training MSE | AIC | Test Set RMSE |
| Early Stopping Techniques | | | | | | | | | |
| LMA | 0.014 | -772.7 | 0.10 | 0.032 | -586.89 | 0.15 | 0.003 | -5952.4 | 0.06 |
| QNA-BFGS | 0.061 | -420.4 | 0.28 | 0.021 | -699.11 | 0.13 | 0.001 | -7112.6 | 0.08 |
| CGA-FR | 0.122 | -254.6 | 0.33 | 0.082 | -322.58 | 0.15 | 0.003 | -5867.9 | 0.08 |
| SCGA | 0.228 | -104.6 | 0.38 | 0.052 | -451.28 | 0.09 | 0.004 | -5952.4 | 0.09 |
| Regularization Technique | | | | | | | | | |
| L2 | 0.275 | -59.83 | 0.42 | 0.123 | -209.62 | 0.18 | 0.0001 | -9544.1 | 0.05 |
| Bayesian | 7.1E-4 | -1492 | 0.07 | 6.0E-5 | -2344.9 | 0.08 | 8.5E-6 | -12147.2 | 0.03 |
| Proposed Technique | | | | | | | | | |
| SSD-LMA | 2.9E-8 | -3911 | 0.06 | 9.6E-9 | -4791.4 | 0.06 | 7.3E-7 | -14729.2 | 0.03 |
| LMA: Levenberg Marquardt Algorithm with Back propagation  QNA-BFGS: Quasi Newton Algorithm (BFGS Update) with Back propagation  SCGA: Scaled Conjugate Gradient Algorithm with Back propagation  CGA-FR: Conjugate Gradient Algorithm (Fletcher Reeves Update) with Back propagation  L2: L2 Regularization on QNA-BFGS with Back propagation  Bayesian: Bayesian Regularization with Back propagation  SSD: Proposed Sample Size Determination Algorithm and LMA | | | | | | | | | |