# Machine Learning Methods in Geoscience

Gerard T. Schuster with labs by Yuqing Chen, Yongxiang Shi, Shihang Feng, Zhaolun Liu, Zongcai Feng, Yunsong Huang, Yan Yang and Tushar Gautam

March 14, 2022

# Contents

**Preface** xv

**Notation Convention** xxiii

**Abbreviations** xxvii

**1 Introduction to ML in Geosciences** **1**

1.1 Introduction . . . 1

1.2 Three Classes of Machine Learning . . . 5

1.2.1 Supervised Learning . . . 5

1.2.2 Unsupervised Learning . . . 7

1.2.3 Reinforcement Learning . . . 7

1.3 Summary . . . 9

**I Mathematical Background** **13**

**2 Data Prediction by Least Squares Inversion** **15**

2.1 Least Squares Inversion . . . 15

2.1.1 Overdetermined, Inconsistent, and Ill-conditioned Equations . . . 16

2.1.2 Least Squares Solution . . . 16

2.1.3 Regularized Least Squares Solution . . . 21

2.1.4 Geometrical Interpretation of Gradient $\nabla\epsilon$ . . . 26

2.1.5 Preconditioning . . . 28

2.1.6 Overfitting Data . . . 29

2.1.7 Inclusion of Bias Factor . . . 34

2.2 Steepest Descent Optimization . . . 34

2.3 Non-Linear Optimization . . . 36

2.3.1 MATLAB Examples of Newton's Method . . . 37

2.4 Summary . . . 41

2.5 Exercises . . . 42

2.6 Appendix: Exact Step Length . . . 46

**3 Specialized Gradient Descent Methods** **49**
3.1 Ad Hoc Gradient Descent Methods . . . 49
3.1.1 Momentum Gradient Descent . . . 49
3.1.2 Adagrad Gradient Descent . . . 56
3.1.3 Adadelta Gradient Descent . . . 58
3.1.4 Nesterov Accelerated Gradient (NAG) . . . 59
3.1.5 Adam Gradient Descent . . . 60
3.1.6 AdaMax Gradient Descent . . . 63
3.1.7 Hybrid Step-length Strategy . . . 63
3.2 Numerical Line Search . . . 64
3.3 Conjugate Gradient . . . 64
3.3.1 MATLAB Conjugate Gradient Code . . . 67
3.4 Summary . . . 70
3.5 Exercises . . . 71
3.6 Appendix: Rosenbrock Plotting Code . . . 73
3.7 Appendix: Step Length Plotting Codes . . . 73

**II Supervised Learning** **75**

**4 Introduction to Neural Networks** **77**
4.1 Introduction . . . 78
4.1.1 Biological Inspiration of a Neural Network . . . 80
4.1.2 General Neural Network . . . 83
4.1.3 Geometrical Interpretation of a Neural Network . . . 86
4.1.4 Generality of a Neural Network . . . 88
4.1.5 Neural Network Workflow . . . 91
4.1.6 Recursive Feed-forward Formula . . . 91
4.1.7 Recursive Back-propagation Formula . . . 93
4.2 Activation Functions . . . 95
4.3 Examples of Neural Networks . . . 98
4.3.1 Single-Layer Neural Network . . . 99
4.3.2 Two-layer Neural Network . . . 100
4.3.3 Neural Network MATLAB Code . . . 103
4.3.4 NN with a Cross-Entropy Objective Function . . . 104
4.4 Types of Classification . . . 111
4.4.1 Multilabel Classification . . . 111
4.4.2 Multiclass Classification . . . 111
4.5 Best Practices . . . 114
4.6 Summary . . . 121
4.7 Exercises . . . 124
4.8 Computational Labs . . . 132
4.9 Appendix: Neural Network Workflow . . . 132
4.10 Appendix: MATLAB Code for Validation . . . 135

**5 Processing Data with FCNN** **137**
5.1 FCNN Filtering of Migration Images in $\mathbf{z} = (x, z)$ . . . 137
5.2 Filtering Angle-Domain Images . . . 141
5.2.1 Angle Domain Transform . . . 141
5.2.2 Filtering Marine Data . . . 143
5.3 NN Filtering of Surface Waves . . . 144
5.4 MATLAB Codes . . . 147
5.5 Summary . . . 149
5.6 Exercises . . . 150
5.7 Computational Labs . . . 150
5.8 Appendix: skeletonized Characteristics in an Image . . . 150
5.9 Appendix: MATLAB Code for Logistic Regression . . . 151

**6 Multilayer FCNN** **155**
6.1 Introduction . . . 155
6.2 Feed-forward Operation . . . 155
6.3 Back-propagation Operation . . . 156
6.3.1 Formula for $\partial\epsilon/\partial w_{jk}^{[N]}$ . . . 157
6.3.2 Formula for $\partial\epsilon/\partial w_{ij}^{[N-1]}$ . . . 157
6.3.3 Formula for $\partial\epsilon/\partial w_{ij}^{[N-2]}$ . . . 158
6.4 MATLAB Code . . . 160
6.5 Summary . . . 162
6.6 Exercises . . . 163
6.7 Computational Labs . . . 163
6.8 Appendix: Vectorized Steepest Descent Formula . . . 163

**7 Support Vector Machines** **165**
7.1 Introduction . . . 165
7.2 SVM Theory . . . 166
7.2.1 SVM Applications . . . 167
7.3 Linear SVM . . . 169
7.4 Nonlinear SVM . . . 172
7.5 Primal and Dual Solutions . . . 174
7.6 Kernel Methods . . . 177
7.7 Numerical Examples . . . 181
7.7.1 MATLAB Codes . . . 182
7.8 Multiclass SVM . . . 183
7.9 Hinge-Loss and Soft-Margin SVM . . . 183
7.10 SVM Lab with CoLab . . . 187
7.11 SVM Regression . . . 192
7.12 Practical Issues for Implementing SVM . . . 192
7.12.1 Scaling . . . 193
7.12.2 Data Augmentation . . . 193
7.12.3 Computational Cost . . . 193
7.12.4 Procedures for Users . . . 195

7.13 Summary . . . . . . . . . . 195
7.14 Exercises . . . . . . . . . . 197
7.15 Computational Labs . . . . . . . . . . 197
7.16 Appendix: Defining the Dual Problem with a Lagrangian . . . . . . . . . . 197
7.16.1 Simple Example of a Dual Solution . . . . . . . . . . 200
7.16.2 Concavity and Maximization of the Reduced Lagrangian . . . . . . . 202
7.17 Appendix: Karush-Kuhn-Tucker Conditions . . . . . . . . . . 203

**III Convolutional Neural Networks 205**

**8 Convolution and Correlation 207**
8.1 Convolution . . . . . . . . . . 207
8.1.1 Time or Shift Invariance . . . . . . . . . . 208
8.1.2 1D Convolution . . . . . . . . . . 210
8.1.3 2D Convolution . . . . . . . . . . 212
8.1.4 3D Convolution . . . . . . . . . . 213
8.2 Correlation and Matched Filters . . . . . . . . . . 213
8.2.1 Multilayer and Multifilter Convolution . . . . . . . . . . 217
8.3 Summary . . . . . . . . . . 220
8.4 Exercises . . . . . . . . . . 220

**9 Convolutional Neural Networks 223**
9.1 Introduction . . . . . . . . . . 223
9.1.1 Brief History of CNN . . . . . . . . . . 224
9.2 Building Blocks of CNN . . . . . . . . . . 227
9.2.1 CNN Forward Modeling . . . . . . . . . . 228
9.2.2 Model Update by Back-propagation . . . . . . . . . . 233
9.2.3 Keras Code . . . . . . . . . . 235
9.2.4 Visualizing the Filters as Correlation Functions . . . . . . . . . . 238
9.2.5 Visualizing the Images Associated with Feature Maps . . . . . . . . 240
9.2.6 Regularization . . . . . . . . . . 242
9.2.7 Mini-Batch, Step Length, Training and Data Augmentation . . . . . 246
9.3 Architectures of CNN . . . . . . . . . . 247
9.3.1 AlexNet . . . . . . . . . . 247
9.3.2 VGGNet . . . . . . . . . . 248
9.3.3 ResNet . . . . . . . . . . 248
9.4 Deep Learning Software . . . . . . . . . . 250
9.5 Seismic Fault Detection by CNN . . . . . . . . . . 251
9.6 Summary . . . . . . . . . . 253
9.7 Exercises . . . . . . . . . . 254

**10 Object Identification by CNN 257**
10.1 Introduction . . . . . . . . . . 257
10.2 Image Classification . . . . . . . . . . 258
10.2.1 Global Picture-size Classification . . . . . . . . . . 258

10.2.2 Localize and Classify a Single Object . . . . . . . . . . . . . . . . . . 258
10.2.3 Judging the Accuracy of the Predicted BB: IOU . . . . . . . . . . . 262
10.2.4 Localize and Classify Multiple Objects . . . . . . . . . . . . . . . . . 262
10.2.5 Pixel-size Classification: Semantic Segmentation . . . . . . . . . . . 270
10.3 Labeling Tools . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 270
10.4 Salt Body Detection by AlexNet . . . . . . . . . . . . . . . . . . . . . . 272
10.4.1 AlexNet CNN Architecture . . . . . . . . . . . . . . . . . . . . . . . 273
10.4.2 Numerical Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 273
10.4.3 Monoparameter CNN . . . . . . . . . . . . . . . . . . . . . . . . . . 273
10.4.4 Multiparameter CNN . . . . . . . . . . . . . . . . . . . . . . . . . . 274
10.5 Detection of Rock Cracks by AlexNet . . . . . . . . . . . . . . . . . . . 274
10.5.1 Unbalanced Crack Data . . . . . . . . . . . . . . . . . . . . . . . . 276
10.5.2 Numerical Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 276
10.6 Detection of Basalt in GPR Images . . . . . . . . . . . . . . . . . . . . . 278
10.6.1 Numerical Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 281
10.7 Fault Detection and Deconvolution . . . . . . . . . . . . . . . . . . . . . 283
10.8 Salt Boundary Detection by U-Net . . . . . . . . . . . . . . . . . . . . . 286
10.8.1 Keras U-Net Code . . . . . . . . . . . . . . . . . . . . . . . . . . . 290
10.8.2 Numerical Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 293
10.9 Fault Detection by U-Net . . . . . . . . . . . . . . . . . . . . . . . . . . 298
10.10 R-CNN Labeling of Blood Cells . . . . . . . . . . . . . . . . . . . . . . 298
10.11 Small Object Detection by Superpixel-based CNN . . . . . . . . . . . . 301
10.11.1 Superpixel Convolution Method . . . . . . . . . . . . . . . . . . . . 302
10.11.2 Numerical Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 303
10.12 Super-resolution Image Enhancement by CNN . . . . . . . . . . . . . . 307
10.13 Identification of Bird Types by CNN . . . . . . . . . . . . . . . . . . . 308
10.13.1 Data Description and AlexNet model . . . . . . . . . . . . . . . . . 309
10.13.2 Numerical Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 309
10.14 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 310
10.15 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 310
10.16 Computational Labs . . . . . . . . . . . . . . . . . . . . . . . . . . . . 312
10.17 Appendix: Selective Search . . . . . . . . . . . . . . . . . . . . . . . . 312
10.18 Appendix: MATLAB Code for AlexNet Crack Detection . . . . . . . . . . 314
10.19 Appendix: Simple Linear Iterative Clustering (SLIC) . . . . . . . . . . . 316

**11 Semantic Segmentation of Cracks in Photos 323**
11.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 323
11.2 Detection and Labeling of Cracks by U-Net . . . . . . . . . . . . . . . . 323
11.2.1 Labeling Cracks . . . . . . . . . . . . . . . . . . . . . . . . . . . . 324
11.2.2 Phase I Training and Validation . . . . . . . . . . . . . . . . . . . . 325
11.2.3 Phase I Testing . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 327
11.2.4 Phase II Training and Validation . . . . . . . . . . . . . . . . . . . . 328
11.2.5 Phase II Testing . . . . . . . . . . . . . . . . . . . . . . . . . . . . 328
11.3 Transfer Training . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 332
11.4 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 333

11.5 Computational Labs . . . 333

**12 Recurrent Neural Networks 337**

12.1 Theory of Recurrent Neural Networks . . . 338
12.1.1 Vanilla RNN Workflow . . . 338
12.1.2 Vanishing Gradient of VRNN . . . 344
12.2 Gated Recurrent Unit Network . . . 348
12.3 Training RNNs . . . 351
12.3.1 Truncated Back-propagation . . . 351
12.3.2 Bidirectional RNN . . . 351
12.3.3 Deep RNN . . . 352
12.4 RNN Applications in Geoscience . . . 354
12.4.1 Predicting Missing Well Logs by BRNN . . . 354
12.4.2 Predicting Well Logs from Seismic Reflection Data . . . 354
12.5 Summary . . . 356
12.6 Exercises . . . 358
12.7 Computational Labs . . . 358
12.8 Appendix: Vanishing Gradients . . . 358
12.9 Appendix: Long Short-Term Memory Architecture . . . 361

**13 Self-attention and Transformers 367**

13.1 Attention . . . 367
13.2 Trained Attention . . . 370
13.2.1 Single- and Multi-head Attention Architectures . . . 374
13.3 Transformer Architecture and Training . . . 378
13.4 Advances in Transformer Models . . . 379
13.4.1 Transformer Models and Event Picking in Seismograms . . . 380
13.5 Summary . . . 383
13.6 Exercises . . . 384

**IV Unsupervised Learning 385**

**14 Autoencoders 387**

14.1 Theory of Autoencoders . . . 390
14.1.1 Linear Autoencoder and PCA . . . 393
14.2 Problems with Autoencoders . . . 394
14.3 Regularization of Autoencoders . . . 395
14.3.1 Dropout . . . 397
14.3.2 Denoising Autoencoders . . . 397
14.3.3 Contractive Autoencoders . . . 398
14.3.4 Sparse Autoencoders . . . 399
14.3.5 Variational Autoencoder . . . 400
14.3.6 Adversarially Constrained Autoencoder Interpolation . . . 404
14.4 Numerical Examples . . . 406
14.4.1 Seismic Interpolation Example . . . 406

14.4.2 Seismic Tomography . . . . . 407
14.4.3 4D Seismic Image Monitoring and Forecasting . . . . . 411
14.5 Summary . . . . . 414
14.6 Exercises . . . . . 415
14.7 Appendix: Autoencoder in Keras . . . . . 416
14.8 Appendix: Dropout=Regularization . . . . . 419
14.8.1 Least Squares Minimization with Dropout . . . . . 420
14.9 Appendix: KL Regularization . . . . . 421

**15 Convolutional Sparse Coding 423**
15.1 Introduction . . . . . 423
15.2 Theory . . . . . 424
15.2.1 Generalization of the Objective Function . . . . . 425
15.2.2 Workflow . . . . . 426
15.3 Numerical Examples . . . . . 427
15.3.1 Synthetic Test . . . . . 427
15.3.2 Field Data Test . . . . . 428
15.4 Discussion . . . . . 430
15.5 Summary . . . . . 434
15.6 Computational Labs . . . . . 438
15.7 Appendix: ADMM . . . . . 438

**16 Principal Component Analysis 441**
16.1 Introduction . . . . . 441
16.1.1 Historical Background of PCA . . . . . 444
16.2 Theory of Principal Component Analysis . . . . . 446
16.2.1 Intuitive Dinosaur Example . . . . . 448
16.2.2 Simple Two-Point Example . . . . . 448
16.2.3 Filtering the PCA Components . . . . . 450
16.2.4 PCA Workflow . . . . . 451
16.2.5 Singular Value Decomposition . . . . . 453
16.3 Numerical Examples . . . . . 455
16.3.1 Yellowstone Eruption Data . . . . . 455
16.3.2 Biplots and Geochemical Data from a Gold Site . . . . . 458
16.3.3 Nashville Limestones . . . . . 458
16.3.4 PCA Applied to Hyperspectral Images . . . . . 460
16.4 Kernel PCA . . . . . 466
16.5 Computational Costs . . . . . 467
16.6 Summary . . . . . 468
16.7 Exercises . . . . . 469
16.8 Computational Labs . . . . . 472
16.9 Appendix: Mathematics of Kernel PCA . . . . . 473
16.10 Appendix: MATLAB Kernel PCA CoLab . . . . . 475

**17 Clustering Algorithms** **479**
17.1 Introduction . . . 479
17.2 K-means Clustering . . . 482
17.2.1 K-means Cluster Lab with Colab . . . 485
17.3 Agglomerative Clustering . . . 486
17.4 Information Weighted Clustering . . . 491
17.5 Fuzzy Clustering . . . 492
17.5.1 Arrival Time Picking and Fuzzy Clustering . . . 493
17.6 Elbow Method for Determining the Number of Clusters . . . 497
17.7 Support Vector Clustering . . . 499
17.8 DBSCAN . . . 502
17.8.1 DBSCAN Lab with Colab . . . 506
17.9 K-Nearest Neighbors Algorithm . . . 507
17.10 Self-organizing Map . . . 508
17.10.1 MATLAB SOM Examples . . . 512
17.11 Numerical Examples . . . 516
17.11.1 Picking Stacking Velocities by K-means Cluster Analysis . . . 516
17.11.2 Generating Models with Sharp Impedance Contrasts . . . 522
17.11.3 DBSCAN Applied to Earthquakes . . . 524
17.11.4 Self-Organized Maps and Classification of Porosity in Rocks . . . 532
17.11.5 Self-Organized Maps and Classification of Seismic Horizons . . . 538
17.11.6 Self-Organized Maps and Fault Detection . . . 539
17.12 Summary . . . 539
17.13 Exercises . . . 540
17.14 Computational Labs . . . 541
17.15 Appendix: Derivation of the Fuzzy Clustering Weight . . . 541

**18 Generative Adversarial Networks** **545**
18.1 Introduction . . . 545
18.2 Theory of GANs . . . 547
18.2.1 GAN Objective Function and Minimax Algorithm . . . 548
18.3 Pseudo-Code for a GAN . . . 550
18.4 Simple Fault Example . . . 555
18.5 GAN Examples in Geoscience . . . 557
18.5.1 Seismic-to-Attribute Sections by a GAN . . . 557
18.5.2 Seismic Resolution Enhancement by GANs . . . 559
18.6 Summary . . . 559
18.7 Exercises . . . 562
18.8 Appendix: GAN Examples . . . 563

**V Bayesian Analysis** **567**

**19 Sampling a Probability Distribution** **569**
19.1 Sampling a Probability Distribution . . . 569

19.2 Sampling a PDF by the Inverse-Transform . . . . . . . . . . . . . . . . . . . . 572
19.3 Sampling a PDF by Acceptance-Rejection . . . . . . . . . . . . . . . . . . . 572
19.4 Markov Chain . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 576
19.5 Metropolis Markov Chain Monte Carlo Sampling . . . . . . . . . . . . . . . 578
19.5.1 Detailed Balance Equation . . . . . . . . . . . . . . . . . . . . . . . 580
19.5.2 Metropolis Sampling Example . . . . . . . . . . . . . . . . . . . . . . 581
19.6 Gibbs Sampling . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 583
19.7 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 587
19.8 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 588
19.9 Appendix: MATLAB Functions for Metropolis Code . . . . . . . . . . . . . 588

**20 Bayes' Theorem 591**
20.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 591
20.2 Intuitive Interpretation of Bayes' Theorem . . . . . . . . . . . . . . . . . 594
20.2.1 Geometrical Interpretation of $P(C), P(T), \frac{P(C)}{P(T)}$ . . . . . . . . . . . 595
20.2.2 Medical Interpretation of $P(C), P(T), \frac{P(C)}{P(T)}$. . . . . . . . . . . . 596
20.2.3 Updating Predictions with New Information . . . . . . . . . . . . . . 598
20.3 Naive Bayes' Theorem . . . . . . . . . . . . . . . . . . . . . . . . . . . . 599
20.4 Motivation for Bayes' Theorem . . . . . . . . . . . . . . . . . . . . . . . 599
20.5 BI and Regularized Least Squares Inversion . . . . . . . . . . . . . . . . . 600
20.6 Geophysical Examples of Bayesian Inversion . . . . . . . . . . . . . . . . . 602
20.6.1 Prediction of an Oil-Spill Source . . . . . . . . . . . . . . . . . . . . 602
20.6.2 VSP Traveltime Inversion . . . . . . . . . . . . . . . . . . . . . . . . 605
20.6.3 Imaging Point Sources in a Hydrofrac Experiment . . . . . . . . . . 613
20.6.4 Rock Types and Uncertainty from Well Log Data . . . . . . . . . . . 617
20.6.5 Rock Types from Well Logs and Seismic Data . . . . . . . . . . . . . 624
20.7 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 624
20.8 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 628
20.9 Appendix: Bayes' Theorem Examples . . . . . . . . . . . . . . . . . . . . 630
20.9.1 Example: Rainy Weather . . . . . . . . . . . . . . . . . . . . . . . . 630
20.9.2 Example: Biking and Weather . . . . . . . . . . . . . . . . . . . . . 632
20.10 Appendix: Naive Bayes' Theorem, Weather and Biking . . . . . . . . . . 634
20.11 Appendix: Kullback-Leibler Regularizer . . . . . . . . . . . . . . . . . . 634
20.12 Appendix: Multisource VSP and $[\mathbf{L}^T\mathbf{L}]^{-1}$ . . . . . . . . . . . . . . . . . 637
20.13 Appendix: Conditional Covariance Matrix . . . . . . . . . . . . . . . . . 638
20.14 Appendix: Bayes and Conditional Covariance Matrix . . . . . . . . . . . . 640
20.15 Appendix: Estimation of Statistical Parameters for VSP Data . . . . . . . 643
20.15.1 Estimating $p(\mathbf{d})$ and $p(\mathbf{m})$ . . . . . . . . . . . . . . . . . . . . . . . 643
20.15.2 Estimating $p(\mathbf{d}|\mathbf{m})$ . . . . . . . . . . . . . . . . . . . . . . . . . . 644

**21 Discriminant Analysis and Gaussian Mixture Model 647**
21.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 647
21.2 Generative and Discriminative Models . . . . . . . . . . . . . . . . . . . . 649
21.3 Linear and Gaussian Discriminant Analysis . . . . . . . . . . . . . . . . . 650
21.3.1 Linear Discriminant Analysis . . . . . . . . . . . . . . . . . . . . . . 651

21.3.2 Gaussian Discriminant Analysis . . . . . 653
21.4 Theory of the Gaussian Mixture Model . . . . . 657
21.4.1 Responsibility Function $p(k|\mathbf{x}^{(n)})$ . . . . . 659
21.4.2 Expectation-Maximization Method . . . . . 660
21.4.3 1D Example of a Responsibility Function . . . . . 661
21.4.4 Geophysical Inversion . . . . . 663
21.4.5 Example of Generative and Discriminative Modeling . . . . . 663
21.4.6 Fuzzy Clustering and GMM . . . . . 665
21.5 Numerical Examples of GMM in Geoscience . . . . . 665
21.5.1 Turbidite Identification from Seismic Data . . . . . 665
21.5.2 Gravity and Magnetic Inversion with GMM Constraints . . . . . 667
21.5.3 Joint Inversion with Updated GMM Penalty Terms . . . . . 671
21.6 Summary . . . . . 672
21.7 Exercises . . . . . 674
21.8 Computational Labs . . . . . 676
21.9 Appendix: Gradients of the GDA Log-likelihood Function . . . . . 676
21.9.1 Solving for $\phi$ . . . . . 677
21.9.2 Solving for $\boldsymbol{\mu}$ . . . . . 678
21.9.3 Solving for $\boldsymbol{\Sigma}$ . . . . . 679
21.9.4 Gradient of the GMM Log-likelihood Function . . . . . 679

**VI Seismic Inversion and Machine Learning** **681**

**22 Physics-Informed Machine Learning Inversion** **683**
22.1 Deterministic and Stochastic Inversion . . . . . 683
22.2 Incomplete Training and Physics-Informed ML . . . . . 685
22.2.1 Data Skeletonization and ML . . . . . 685
22.3 Physics-Informed Machine Learning . . . . . 686
22.3.1 Sequential ML+Physics Inversion . . . . . 687
22.3.2 Simultaneous ML+Physics Inversion . . . . . 687
22.4 Automatic Differentiation . . . . . 687
22.5 Summary . . . . . 687

**23 Tomographic Deconvolution** **689**
23.1 Introduction . . . . . 689
23.1.1 Previous Work . . . . . 690
23.2 Theory of Tomographic Deconvolution . . . . . 691
23.2.1 Migration Deconvolution versus Tomographic Deconvolution . . . . . 692
23.2.2 Tomographic Deconvolution Workflow . . . . . 692
23.2.3 CNN Architecture . . . . . 693
23.3 Numerical Results . . . . . 694
23.3.1 Reflections: Single Anomaly Models . . . . . 694
23.3.2 Reflections: Multiple Anomaly Models . . . . . 695
23.3.3 Models that Break the Deconvolution Filter . . . . . 695

23.3.4 Refractions: Multiple Anomaly Models . . . . . . . . . . . . . . . . . 699
23.3.5 Refractions: Fault Anomaly Models . . . . . . . . . . . . . . . . . 699
23.4 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 700

**24 Neural Network Least Squares Migration 703**
24.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 703
24.2 Theory of Neural Network Least Squares Migration . . . . . . . . . . . . . . 705
24.2.1 Least Squares Migration . . . . . . . . . . . . . . . . . . . . . . . 705
24.2.2 Sparse Least Squares Migration . . . . . . . . . . . . . . . . . . . . 706
24.2.3 Neural Network Least Squares Migration . . . . . . . . . . . . . . . 707
24.2.4 Multilayer Neural Network LSM . . . . . . . . . . . . . . . . . . . 709
24.3 Numerical Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 709
24.3.1 Three-layer Velocity Model . . . . . . . . . . . . . . . . . . . . . . 710
24.3.2 SEG/EAGE Salt Model . . . . . . . . . . . . . . . . . . . . . . . . 712
24.3.3 North Sea Data . . . . . . . . . . . . . . . . . . . . . . . . . . . . 714
24.4 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 718
24.5 Conclusions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 721
24.6 Appendix: Migration Green's Function . . . . . . . . . . . . . . . . . . . 721
24.7 Appendix: Soft Thresholding Function . . . . . . . . . . . . . . . . . . . 722
24.8 Appendix: Neural Network Least Squares Migration . . . . . . . . . . . . . 723
24.9 Appendix: Alignment of the Filters . . . . . . . . . . . . . . . . . . . . . 724

**25 Wave Equation Inversion and Neural Networks 729**
25.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 729
25.2 Theory for Wave Equation Inversion of Skeletonized Data . . . . . . . . . . 731
25.2.1 Feature Extraction . . . . . . . . . . . . . . . . . . . . . . . . . . 732
25.3 NML Theory . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 733
25.4 NML Inversion of Crosswell Data . . . . . . . . . . . . . . . . . . . . . . 735
25.4.1 Assessing the Optimal Dimension of the Latent Space . . . . . . . . 736
25.5 NML Inversion of Refraction Data . . . . . . . . . . . . . . . . . . . . . 737
25.6 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 742
25.7 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 743
25.8 Connective Function . . . . . . . . . . . . . . . . . . . . . . . . . . . . 744
25.8.1 Gradient . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 745

**26 Automatic Differentiation 747**
26.0.1 Automatic Differentiation Algorithm . . . . . . . . . . . . . . . . . 747
26.1 Modeling, Automatic Differentiation and Inversion . . . . . . . . . . . . . 753
26.1.1 Automatic Differentiation for the One-way Wave Equation . . . . . 753
26.1.2 Automatic Differentiation and Waveform Inversion . . . . . . . . . . 754
26.2 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 756
26.3 Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 756
26.4 FD Solution to the Wave Equation . . . . . . . . . . . . . . . . . . . . . 757

**VII Background Appendices** **763**

**27 Appendix: Geophysics Background** **765**
27.1 Seismic Data Recording and Processing . . . . . 765
27.2 Diffraction Stack Migration . . . . . 772
27.3 Surface Waves . . . . . 776
27.3.1 Estimating Depth vs Frequency . . . . . 777
27.4 Seismic Reflection Data as an LTI System: $s(t) = r(t) \star w(t)$ . . . . . 777

**28 Probability Theory for Machine Learning** **783**
28.1 Random Variables, Sample Space, Events and Probability . . . . . 783
28.1.1 Types of Events . . . . . 784
28.1.2 Probability . . . . . 785
28.1.3 Law of Total Probability . . . . . 790
28.2 Bayes' Theorem . . . . . 793
28.2.1 Bayes' Theorem and Cancer Example . . . . . 793
28.2.2 Bayes Theorem for $K$ Disjoint Events . . . . . 794
28.2.3 Bayesian Terminology . . . . . 794
28.3 Gaussian Probability Distribution . . . . . 795
28.3.1 Central Limit Theorem . . . . . 796
28.3.2 Multivariate Normal Distribution . . . . . 796
28.4 Sampling a Probability Distribution . . . . . 798
28.5 Maximum Likelihood Estimate and Neural Network Models . . . . . 798
28.6 Summary . . . . . 800
28.7 Exercises . . . . . 800

# Preface

This book presents the theory of machine learning (ML) algorithms and their applications to geoscience problems. More than half of the described algorithms fall under the class of neural network methods. Their description is at a level that can be understood by anyone with a modest background in linear algebra, calculus and probability. An elementary working knowledge of MATLAB is assumed and almost every chapter is accompanied by lab exercises to reinforce the ML principles. If you don't know MATLAB or Python, many of the labs use the CoLab, aka Colaboratory, notebook which is an easy-to-use product from Google that allows for the execution of Keras-based ML computer codes. It also provides for the free use of a cloud-based GPU processor[1] that can be used as soon as you login. There is no need for the troublesome installation of notebook software.

There are many definitions of *machine learning* (ML), but one of the best is attributed to Arthur Samuel[2]:

*Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed.*

This definition doesn't mean we don't have to write the programs, it means that the ML algorithm finds the best model by learning from the data without relying on rules-based programming. For example, finding the best fit model $\mathbf{y} = f(\mathbf{x})$ by a typical least squares procedure typically assumes a linear mathematical model $f(\mathbf{x})$ that predicts the output $\mathbf{y}$ from the input $\mathbf{x}$. For example, $\mathbf{W}\mathbf{x} = \mathbf{y}$, where $\mathbf{W}$ is a matrix. A ML algorithm such as a neural network avoids this assumption by devising the best non-linear model learned from the data $\mathbf{y}$ for the given architecture.

**Neural Network Example.** How does the neural network learn the model[3] $f(\mathbf{x}) = \mathbf{y}$ from the data? Learning the model $f(\mathbf{x})$ is accomplished by using a large training set of data $(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})$ for $n \in \{1, 2, \ldots, N\}$, and adjusting the parameters of the model until $f(\mathbf{x}^{(n)})$ can accurately predict $\mathbf{y}^{(n)}$ for any training pair. Once the model is learned, then it can accurately predict the output $f(\mathbf{x}^{new}) = \mathbf{y}^{new}$ from a new input $\mathbf{x}^{new}$. This can be important

[1]Colab is a hosted Jupyter notebook service that requires no setup to use, while giving free access to computing resources including GPUs. See *https : //colab.research.google.com/notebooks/intro.ipynb*

[2]This quote is often attributed to Arthur Samuel, but it cannot be found in Samuel (1959, 1967). A similar quote can be found at https://www.techemergence.com/what-is-machine-learning/. See http:// infolab.stanford.edu/pub/voy/museum/samuel.html for a brief bio of this pioneering engineer.

[3]"Learning the model" is equivalent to computing the parameters of the non-linear function $f(\mathbf{x})$ that allow for an accurate prediction of $f(\mathbf{x}) = \mathbf{y}^{true}$ for all the training pairs $(\mathbf{x}, \mathbf{y}^{true})$.

for many applications, such as predicting the presence of underground contaminants $\mathbf{y}^{new}$ from processed geoelectrical and/or seismic data $\mathbf{x}^{new}$, accurately detecting the presence of breast cancer $\mathbf{y}^{new}$ recorded on a fusion of CAT scan and MRI images $\mathbf{x}^{new}$, or delineating the faults in $\mathbf{y}^{new}$ on the right-hand side of Figure 1 from the input seismic section $\mathbf{x}^{new}$ on the left-hand side.

The mathematical operations of a convolutional neural network model are represented by the block diagrams in Figure 1, where the input image $\mathbf{x}$ of a seismic section is decomposed into component images contained in the FM1 block (or shallowest layer). Here, a block represents a set of mathematical operations that produce a stack of 2D images known as feature maps (FMs). Each of the 32 feature maps in FM1 is computed by convolving a small filter ($3 \times 3$ filter in this example) with the input image, and then each pixel value in the FM is thresholded to be between 0 and 1 by a non-linear squashing function[4]. The $FM1$ feature maps are then subsampled by a factor of four (aka known as maxpooling), and the resulting images are filtered and thresholded by the FM2 $3 \times 3$ filters to get the 64 FMs for block 2. The subsampling acts as a low-pass spatial filter so that only coarser features of the seismic section are seen in, for example, FM3. In the end, the FMs represent the decomposition of a complicated object into high-wavenumber images on the left and low-wavenumber images on the far right. The FMs are weighted, upsampled, summed together and squashed to make a single decision, i.e. classification, at each pixel associated with the input image (Long et al., 2015). Does the pixel value represent a fault, then the answer is yes $y = 1$ for its class. If not, then the answer is no $y = 0$ for the output class. The displayed FMs are upsampled by a deconvnet procedure to transform them to the same size as the input image.

**Neural Networks and Deeper Connections.** Neural network models lack a rigorous theoretical foundation that explains their formidable number of commercial successes. This has motivated physicists and mathematicians to search for its mathematical and physical foundations. For example, they have recently discovered that successful CNN modeling can have many more tunable parameters than training data, i.e. equations of constraint, without suffering from overfitting. Arora et al. (2018) proves that a large number of FMs can be redundant and allows for simple compression of the model.

Another insight says that the CNN decomposition is similar to a dictionary-learning algorithm (Papayan et al., 2016, 2017a, 2017b; Sulam et al., 2018), where the different FMs can represent the multiscale components of a complicated object, such as its atoms or molecules. As an example, Figure 2 illustrates how two convolutional FMs, denoted as elementary "atoms" in the bottom row, are combined and filtered to create slightly more complex structures, i.e. molecules, at the second level. These "molecules" are then combined by a CNN model to create the global atom representing, in this case, a digit.

However, the basic operations of the neuron, i.e. atom, has much more complexity than previously thought, where the atom itself is composed of even finer structures. Work by Beniaguev et al. (2021) suggests that the cortical neuron mathematically performs the

[4]A squashing function $\sigma(z)$ *squashes* a wide range of input values $-\infty < z < \infty$ to be between a small range of output values such as $0 \leq \sigma(z) \leq 1$. The squashing operation sparsifies essential+extraneous information into its essential components (Papayan et al., 2016, 2017a, 2017b; Sulam et al., 2018; Chen et al., 2020b).

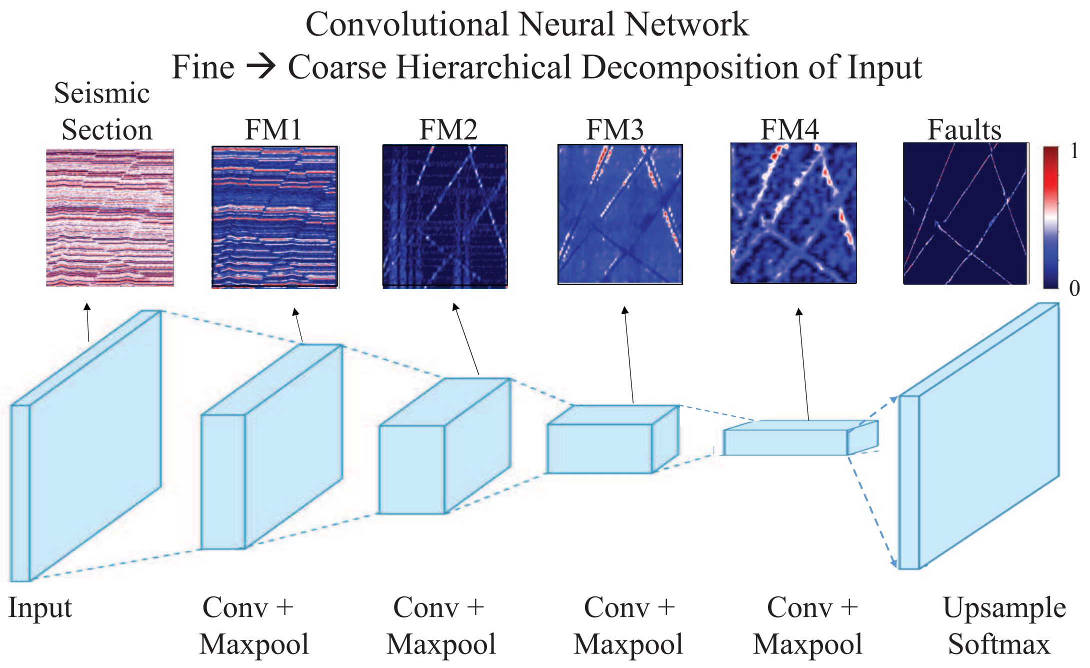


Figure 1: Block diagram of the convolutional neural network (CNN) architecture that decomposes the input picture of a seismic section into fine-scale feature maps (FM1 and FM2) and coarser-scale feature maps (FM3 and FM4). To provide a more interpretable image, the FMs are *deconvolved* for the display of interpretable images (Shi et al., 2018). The pixels in the far-right image are labeled $0 < y < 1$, where $y = 1$ indicates that the pixel is a fault and $0 \leq y \leq 1$ indicate the probability of being a fault. Typically, the shallow $3 \times 3$ filters that compute the 32 FM1 images from the input image detect the sharp edges in the input, and the filters associated with FM2 to FM3 detect larger scale features such as the general shape of the fault. These *successive decompositions derive the building blocks at successive levels from specific combinations of building blocks at the previous level* (Holland, 1995). Figure partially adapted from Dertat (2017) and Shi et al. (2018).

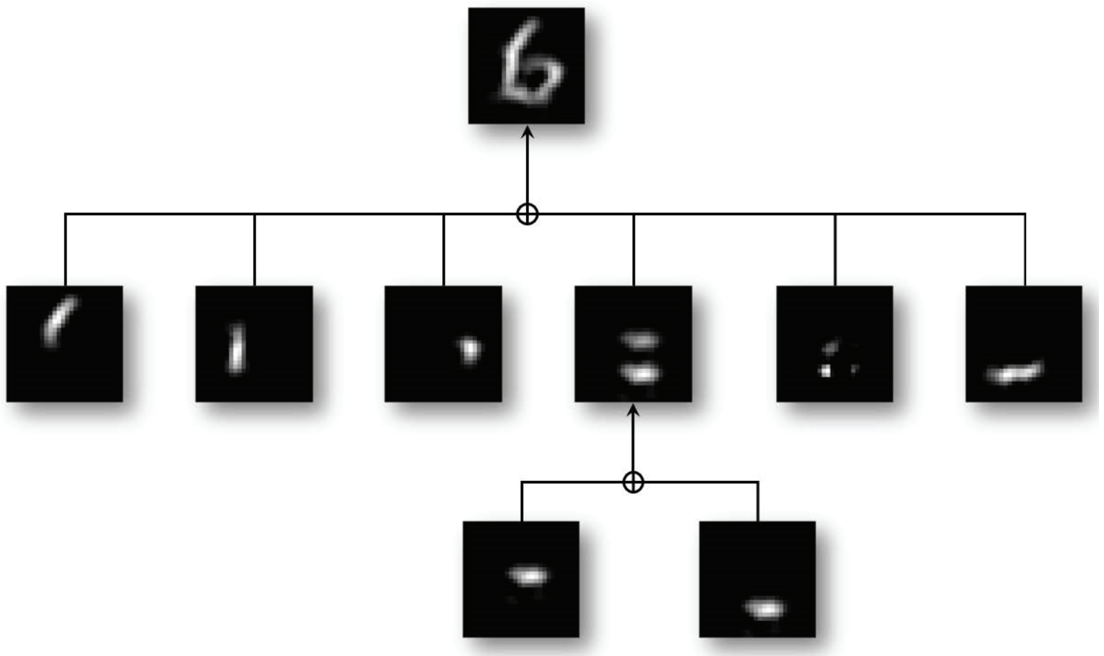


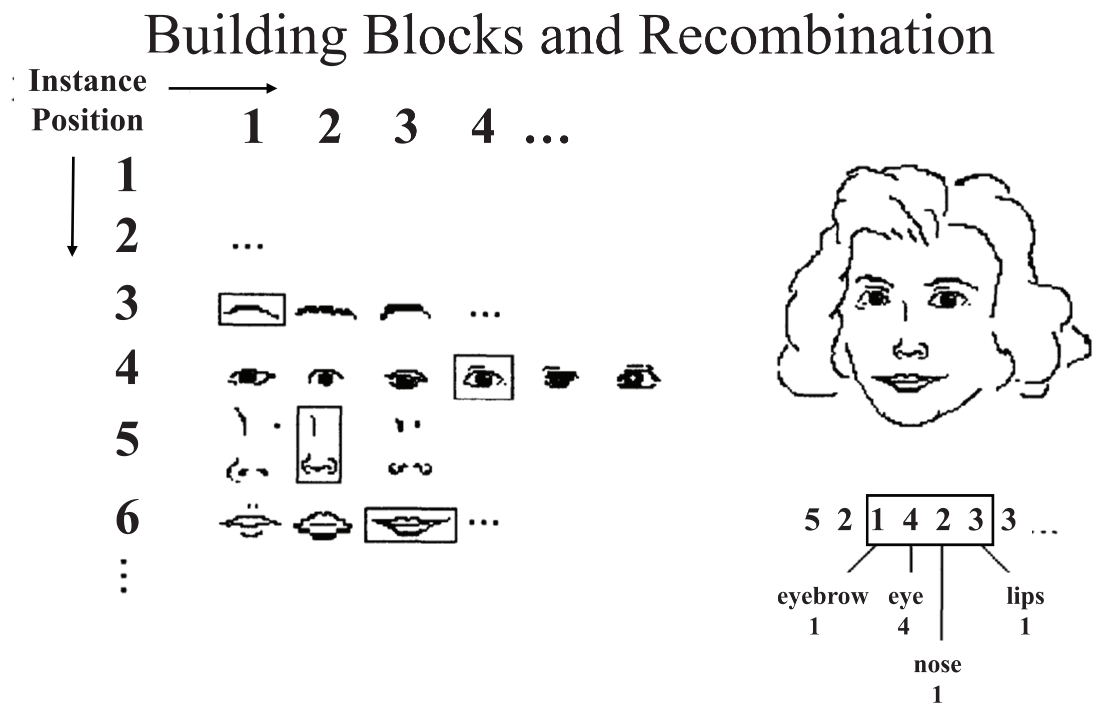
Figure 2: Elementary FMs in the bottom row are combined to form the molecules in the second row, which in turn are used to form the digit 6 in the top row. Figure from Sulam et al. (2018).

same functions as a deep neural network with 5-8 layers. This depth is associated with the neuron's dendritic branches that act as spatial pattern detectors.

Decomposing an input image into a sequence of simpler-to-more complicated parts is also used to describe the process of adaptive evolution illustrated in Figure 3. John Holland (1995) describes the fundamental set of procedures for adaptive evolution in nature:
*'If model making, broadly interpreted, encompasses most of scientific activity, then the search for building blocks becomes the technique for advancing that activity. At a fundamental level, we have the quarks of Gell-Mann. Quarks can be combined to yield nucleons, the building blocks at the next level. The process can be iterated, deriving the building blocks at successive levels from specific combinations of building blocks at the previous level. The result is the quark/ nucleon/ atom/ molecule/ organelle/cell/ underpins much of physical science.'*.

**Artificial Intelligence and CNN.** More generally, ML methods belong to the larger class of artificial intelligence[5] algorithms (see Figure 4), but the one that is attracting the most interest is that of deep learning with CNNs. Convolutional neural networks are now playing leading roles in the software for self-driving cars, language translation, speech recognition, computer vision, fusion and diagnosis of medical images, analysis of satellite and aerial

[5]https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/



A face can be described by stringing together the numbers that index its different component parts.

Figure 3: Neural networks is similar to John Holland's (1995) description of how nature evolves models: '...building blocks at successive levels from specific combinations of building blocks at the previous level.". The important features of the face are denoted by the *feature* rectangles on the left. Figure adapted from Holland (1995).

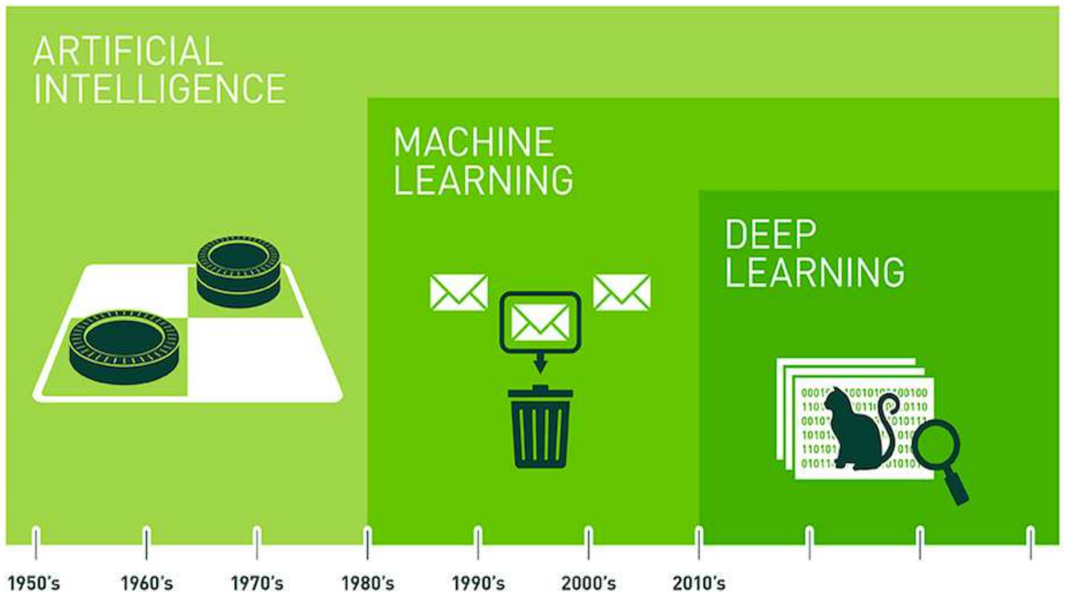


Figure 4: Artificial intelligence (AI) and the sub-classes of machine learning and deep learning, where AI involves a machine doing something that only a human would be able to do. Source of picture is Nvidia.

images for optimizing agricultural practices, and many other fields. Significantly, it is now being recognized as an important interpretation tool for analyzing lithology from seismic and well-log data, detecting faults in large 3D volumes of seismic data, and searching for patterns in earthquake data.

**Organization of the Book.** This book is divided into six main sections:

- Mathematical Background and Optimization Theory. Optimization theory and gradient descent methods for ML are reviewed, with an emphasis on the choice of step length methods. Readers with a familiarity of these algorithms can skip to the next sections.

- Supervised Learning Algorithms. Introduction to neural networks, fully-connected neural networks, applications to geoscience problems, and support vector machines.

- Convolutional Neural Networks. Basics of convolution and correlation, CNN, object identification, semantic segmentation, recurrent neural networks, Transformers and self-attention.

- Unsupervised Learning Algorithms: Autoencoders, convolutional sparse coding, principal component analysis, clustering methods, generative adversarial networks.

- Bayesian Analysis. Sampling, Bayes' theorem, and Gaussian mixture models.

- Seismic Inversion and ML. Physics-informed ML inversion, tomographic deconvolution, neural network LSM, wave equation inversion and neural networks, automatic differentiation.

Most chapters are accompanied by executable code (MATLAB and/or Keras with CoLab) that can be implemented by the reader with a modest level of programming knowledge. The understanding of the algorithms is deepened by parameter tuning and by examining the details of the code. Most of these codes have been written by my co-authors, without which this book would not have been written.

Case histories are used to demonstrate the practical use of ML in solving geoscience problems. These problems include fracture identification in photos, detection of rock cracks in drone photos, fossil and lithology detection in thin sections, prediction of permeability in rock samples, geochemical analysis, well-log analysis and identification of salt boundaries in seismic data, seismic arrival-time picking by a clustering method, least squares migration, velocity analysis, demultiple, noise reduction in seismic data and migration images, and some interpretation examples for seismic and radar data. The reader who diligently goes through the chapters and labs will have a thorough grounding in some of the fundamental ML methods used in geoscience.

**WWW Software Sites for Computational Labs.** The computational labs for the book *Machine Learning in Geoscience* are located at the following sites:

- *Machine Learning in Geoscience* Labs: http://csim.kaust.edu.sa/files/ErSE394/LAB1/
- *Machine Learning in Geoscience* Labs: http://utam.gg.utah.edu/books/ML/index.html
- *Machine Learning in Geoscience* Labs: http://seg.org/books/Schuster.ML/index.html
- *Machine Learning in Geoscience* Labs: repository.kaust.edu.sa/bitstream/handle/10754/674007/GeoscienceMachineLearning.html
- *Machine Learning in Geoscience* Labs: https://earth.utah.edu/books/Schuster.ML/index.html

The computational labs for the books *Seismic Interferometry* and *Seismic Inversion* are located at the following sites:

- *Seismic Interferometry* Labs: http://utam.gg.utah.edu/Inter.LAB1/
- *Seismic Inversion* Labs: repository.kaust.edu.sa/bitstream/handle/10754/674016/SeismicInversion.html
- *Seismic Inversion* Labs: https://earth.utah.edu/books/Schuster.SeismicInversion/index.html

## Acknowledgments

GTS is very thankful for the financial and computer support provided by King Abdullah University of Science and Technology and the KSL facilities (https://www.hpc.kaust.edu.sa

/). The assistance of Professor Xiangliang Zhang (KAUST and Notre Dame University), Dr. David Pugh (KSL) and Dr. Saber Feki (KSL) were very beneficial in bringing me to the frontiers of teaching ML. My loyal co-authors are to be acknowledged for writing many of the computer codes that accompany the chapters. I am very grateful to Professor Daniel Trad at Calgary University for his tireless editing of the manuscript. He provided many detailed comments and suggestions for improvement. I am deeply grateful to Dr. Yan Yang for discussions about modifications and her expert help in organizing and implementing corrections, and appreciate the faithful editing of the manuscript by my co-authors. This book benefitted from the inclusion of appendix 17.6.2 written by Yunsong Huang, the excellent geoscience examples provided by Eduardo Cano, Ahmad Ramdani and Tian Qiao, the concise summaries of many ML classes provided by Yuan Yang, and thorough lab instructions by Shi Yongxiang.