1. Course Overview

Gradient descent (GD) is a well-known first order optimization method, which uses the gradient of the loss function, along with a step-size (or learning rate), to iteratively update the solution. When the loss (cost) function is dependent on datasets with large cardinality, such as in cases typically associated with deep learning (DL), GD becomes impractical.

In this scenario, stochastic GD (SGD), which uses a noisy gradient approximation (computed over a random fraction of the dataset), has become crucial. There exits several variants/improvements over the "vanilla" SGD, such as RMSprop, Adagrad, Adadelta, Adam, Nadam, etc., which are usually given as black-boxes by most of DL's libraries (TensorFlow, PyTorch, MXNet, etc.).

The primary objective of this course is to combined the essential theoretical aspects related to SGD and variants, along with hands on experience to program in Python, from scratch (i.e. not based on DL's libraries such as TensorFlow, PyTorch, MXNet) the SGD along with the RMSprop, Adagrad, Adadelta, Adam and Nadam algorithms and to test their performance using the MNIST and CIFAR-10 datasets for shallow networks (consisting of up to two ReLU layers and a Softmax as the last layer).

2. Learning Outcome

- Learn the Bayesian approach to inverse problems.
- Learn about differences between gradient descent (GD) and stochastic GD.
- Learn about SGD variants, as well as on their improvements over the vanilla SGD.
- Learn about the backpropagation algorithm and its use within SGD.
- Gain experience in programming SGD for training shallow networks (consisting of up to two ReLU layers and a Softmax as the last layer).
- Build confidence to take on his/her own project related to machine learning / deep learning.
3. Syllabus and Presenters

Lecture 1:
- Introduction.
- Basic concepts.
  - Bayes' theorem.
  - MAP (maximum a posteriori).
  - Linear regression.
  - Logistic and softmax regression.
  - Gradient descent (GD) and stochastic GD.
- Hands-on 1.1 (see Section 7).

Lecture 2:
- Accelerated GD (AGD).
  - Adaptive step-sizes.
  - Momentum.
  - Nesterov acceleration.
  - Anderson acceleration.
- Hands-on 2.1 (see Section 7)
- SGD variants.
  - AdaGrad.
  - AdaDelta.
  - RMSprop.
  - Adam.
  - Nadam.
- Hands-on 2.2 (see Section 7)

Lecture 3:
- Hidden layers.
  - Linear vs. non-linear.
  - Activation functions.
- Hands-on 3.1. (see Section 7)
- Computing gradients.
- The backpropagation (BP) algorithm.
- SGD and BP working together.
- Hands-on 3.2 (see Section 7)

Lecture 4:
- Hands-on 4.1 (see Section 7)
- Quick overview of deep learning (DL).
  - Convolutional layer.
  - Other layers: maxpool, dropout, dense, etc.
- Hands-on 4.2 – CIFAR10 dataset.

4. Target audience, and the expected prerequisite technical knowledge

The targeted audiences are senior-year undergraduate, postgraduate (specially first year), as well as industry signal processing engineers and practitioners, with some background in python, random signal processing, (basic) optimization and linear algebra.

5. Supporting course resources, software, tools and readings

- Lecture notes from the slides presented in the course.
- Python Jupyter notebooks in Google colab for hands-on practice.
- References to papers for specific details taught in the course

6. Pre-reading:


7. Hands-on or lab components of the short course (For virtual course, please indicate how this would work over Zoom)

- Hands-on 1.1
  - The MNIST dataset
  - Data preparation
  - GD implementation; simple (quadratic) test.
  - SGD implementation; multiclass regression for the MNIST dataset.

- Hands-on 2.1
  - Accelerated GD implementation; comparisons w.r.t. GD.
  - Hands-on 2.2
  - SGD variants implementation.
  - Multiclass regression for the MNIST dataset.

- Hands-on 3.1
  - Impact of adding one, random value, ReLU hidden layer.
  - Classification of the MNIST dataset.
  - Classification of the CIFAR dataset.

- Hands-on 3.2
  - BP and SGD along with one ReLU hidden layer.
  - Classification of the CIFAR dataset.

- Hands-on 4.1
  - BP and SGD along with two ReLU hidden layers.
  - Classification of the CIFAR dataset.

- Hands-on 4.2
  - Using TensorFlow (TF).
  - Performance comparison (w.r.t. Previously developed code).
  - Implementing your own solver in TF.
  - Using simple DL networks.

8. Presenters’ contact information and short biography

<table>
<thead>
<tr>
<th>Presenter</th>
<th>Short Biography</th>
</tr>
</thead>
</table>


| **Paul Rodriguez** | Paul Rodriguez received the BSc degree in electrical engineering from the ‘Pontificia Universidad Católica del Perú’ (PUCP), Lima, Peru, in 1997, and the MSc and PhD degrees in electrical engineering from the University of New Mexico, U.S., in 2003 and 2005 respectively. He spent two years (2005-2007) as a postdoctoral researcher at Los Alamos National Laboratory, and is currently a Full Professor with the Department of Electrical Engineering at PUCP. His research interests include AM-FM models, parallel algorithms, adaptive signal decompositions, and optimization algorithms for inverse problems in signal and image processing such Total Variation, Basis Pursuit, principal component pursuit (a.k.a. robust PCA), convolutional sparse representations, extreme learning machines, etc. |
| prodig@pucp.edu.pe | |
| +511 626 2000 ext 4681 | |
| 9. Recent related publications | Recent related publications |


