JOSE
The Journal of Open Source Education

An Interactive Notebook for Learning Nonlinear Programming with Constraints

Anselm Hudde [1] and **Maren Hackenberg** [2]

1 University of Applied Sciences Koblenz, Germany **2** University of Freiburg, Freiburg

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Submitted: 01 June 2025
Published: unpublished

Summary

This Jupyter notebook introduces the fundamental concepts of nonlinear programming. Using interactive plots, the notebook explains the gradient descent algorithm for optimization, as well as the penalty method for incorporating constraints. It also demonstrates how to apply nonlinear programming using the Python package `SciPy` and concludes with an outlook on automatic differentiation. The notebook can easily be run from any browser via Binder without additional technical prerequisites. It can also be run locally without many dependencies.

Statement of Need

The gradient descent algorithm, introduced by Augustin-Louis Cauchy in 1847 Cauchy (1847), has become one of the most widely used algorithms, with applications ranging from economics (e.g., portfolio optimization Suykens et al. (2014); game theory Daskalakis & Panageas (2018)), to engineering Martins & Ning (2021), operations research Taha (2017), and machine learning Goodfellow et al. (2016); (**?**).

Even though gradient descent is very intuitive once a mental picture of the algorithm is formed, students often struggle to grasp the basic concepts. Considering that, it is even more difficult for students to picture how constraints change the nature of the problem and how the penalization method works.

Often, gradient descent is explained by formulas or 2D plots. Visualizing the single descent steps for a function with values in $\mathbb{R}$ misses the richness of the method in more dimensions, and displaying plots for functions with values in $\mathbb{R}^2$ is always a projection. But the function surface is best understood when one can rotate it in space to view it from different angles—much like turning an object in one's hands or observing a statue from various perspectives.

That is exactly what this Jupyter notebook delivers: The student can actively rotate 3D plots of the function surface to easily grasp its structure. On top of that, the student can choose parameters herselve and progress through the steps interactively.

For this reason, we have developed a Jupyter notebook designed for undergraduate and graduate students from diverse backgrounds to study gradient descent through hands-on exploration.

Furthermore, being a Jupyter notebook with Python code offers the advantage that it can be run from any browser via Binder without additional technical prerequisites, which significantly lowers the activation energy required to start learning. Alternatively, this notebook can also be easily run on the student's computer without relying on external servers, making it easy to maintain.

The Journal of Open Source Education

Description, Instructional Design, and Project Story

This notebook uses interactive contour and surface plots to build intuition for gradients and optimization.

This notebook first motivates gradient descent by giving a practical example of a constrained optimization problem. After that, intuition for the basic concept of a gradient is developed. Then, gradient descent is demonstrated, where each step can be tried out individually. In this way, the user can experience first-hand the role of the step size and the starting point. Next, the use of the Python package `SciPy` for nonlinear programming is explained.

This notebook also covers nonlinear programming with constraints. It begins by developing intuition for various forms of constraints and continues with an explanation and illustration of penalization. It also shows how constraints can be taken into account in the previously introduced `SciPy` nonlinear programming approach.

Since classical gradient descent with analytic or finite-difference derivatives is not suitable for many applications—including deep learning—the final section provides an outlook on automatic differentiation.

This module is regularly used in a data science class for undergraduate students of mathematics but is also suitable for other classes where gradient descent is needed and for students with less mathematical background.

References

Cauchy, A. (1847). Méthode générale pour la résolution de systèmes d'équations simultanées. *Compte Rendu Des Séances de l'académie Des Sciences*, 536–538.

Daskalakis, C., & Panageas, I. (2018). The limit points of (optimistic) gradient descent in min-max optimization. *Neural Information Processing Systems*. https://api.semanticscholar.org/CorpusID:54170344

Goodfellow, I. J., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Martins, J. R. R. A., & Ning, A. (2021). *Engineering design optimization*. Cambridge University Press.

Suykens, J. A. K., Signoretto, M., & Argyriou, A. (2014). *Regularization, optimization, kernels, and support vector machines (1st ed.)*. Chapman; Hall/CRC.

Taha, H. (2017). *Operations Research An Introduction, Global Edition*. Pearson. ISBN: 9781292165547