Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich


# Mathematical Foundations of Computer Graphics and Vision

EXERCISE 1 - ROBUST ESTIMATION AND OPTIMIZATION

Handout date: 28.02.2023
Submission deadline: 14.03.2023, 23:59

## General Rules

**Plagiarism note.** Copying code (either from other students or from external sources) is strictly prohibited! We will be using automatic anti-plagiarism tools, and any violation of this rule will lead to expulsion from the class.

Late submissions up to 24h will be accepted with a penalty, except for extensions in case of serious illness or emergency. In that case please notify the assistant and provide a relevant medical certificate.

**Software.** All exercises of this course use Python. See the exercise session slides and this document for hints or specific functions that could be useful for your implementation.

**What to hand in.** Upload a single .ipynb file of your solution on Moodle. The file must be called "MATHFOUND23-*-firstname-familyname.ipynb" (replace * with the assignment number). Write self-documenting code and comment the more complex parts.

**Grading.** This homework is 8.3% of your final grade. Your submission will be graded according to the quality of the images produced by your program, the conformance of your program to the expected behaviour of the assignment, and your understanding of the underlying techniques used in the assignment. The submitted code must produce exactly the same images included in your submission (up to randomness).

## Goal of this exercise

In this exercise you will apply what you learned about robust optimization, especially RANSAC and Iteratively Reweighted Least Squares (IRLS). You will implement, in Python, RANSAC for polynomial fitting in the presence of outliers, IRLS for line fitting with $L_1$ norm, and LP for line fitting with $L_1$ and $L_\infty$ norms.

Use the Jupyter notebook that is provided with the exercise.

## 1. EXERCISE PART 1: RANSAC for circle fitting

As described in the lectures, RANSAC can be used to fit a model in the presence of outliers and identify these outliers. In this exercise, we will utilize RANSAC for polynomial fitting with different ratios of outliers.

Given a set of $N$ 2D data points corrupted by outliers, the goal is to detect the dominant $n$-th degree polynomial, that is the polynomial passing through the highest number of points, up to an inlier threshold. You will create results similar to Figure 1. For this, you will perform the following tasks:

(1) generate synthetic data sets
(2) implement model fitting
(3) implement and run RANSAC
(4) plot the distribution of the number of inliers
(5) implement and run exhaustive search

In this exercise, we consider polynomial model of the form $y = \Sigma_{i=0}^{n} \alpha_i x^i$ where $\{\alpha_i\}$ are its coefficients. Use the vertical distance between the line and points.

### 1.1. **Data generation and Model Fitting.**

You will generate a set of $N$ 2D data points on the polynomial in the domain $[-10, 10] \times [-10, 10]$. These points are corrupted by (small) noise. For this, add random uniform noise $X \sim \mathcal{U}_{[-0.1,0.1]}$ on the $y$ coordinate of the data points.

Additonally, the data also has outliers - points that don't follow the model at all. Generate a ratio $r$ of outliers with $r = 5\%$, $20\%$, $30\%$ and $70\%$. Define $\tau = 0.1$ as the inlier distance threshold.

The total number of points is always $N$ ($N$=100). If the ratio of outliers is $r = 10\%$, then 10 points are outliers and 90 points are inliers. If $r = 30\%$, then 30 outliers and 70 inliers. When generating the data points, make sure that the synthesized inliers are indeed inliers and the synthesized outliers are indeed outliers and then verify the given outlier ratio $r$.

Finally, implement a function fitting the highest degree polynomial on a given subset of points.

### 1.2. **RANSAC.**

You will implement RANSAC as described in the lecture. Compute the number of iterations with the generated outlier ratio $r$, the minimal number of points and a guaranteed accuracy of 99%.

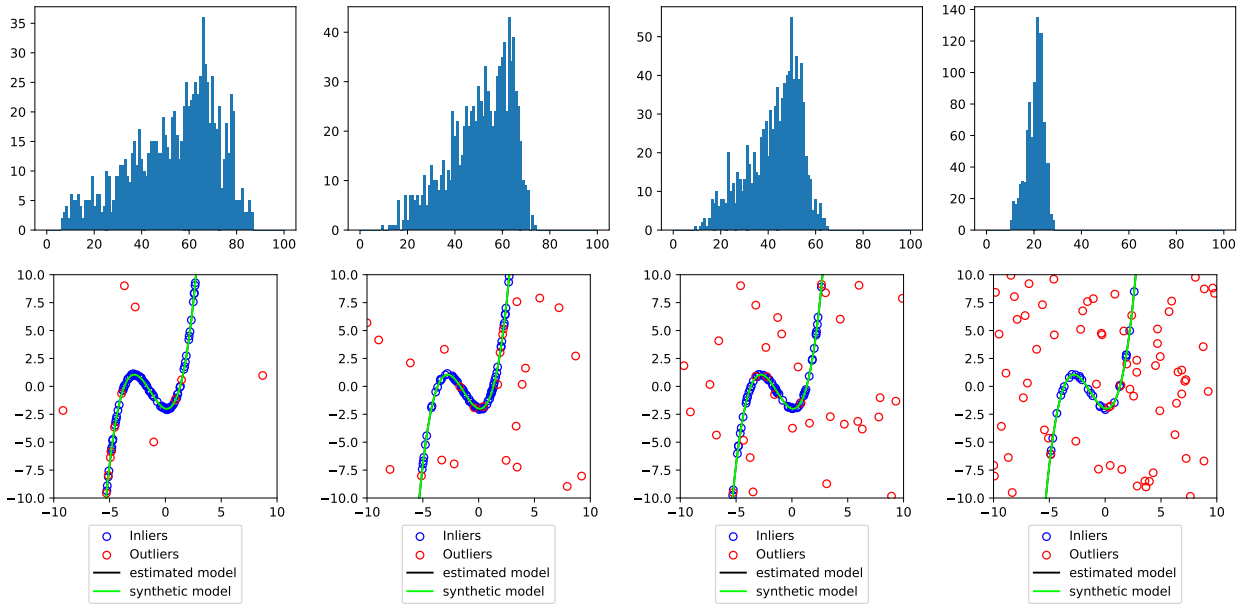


FIGURE 1. Distribution of RANSAC results with different outlier ratios (5%, 20%, 30%, 70%).

1.3. **Exhaustive search.** Implement exhaustive search by trying all the combinations of minimal data points.

A few hints about the implementation:

- We assume you already know basic Python programming. If you need a small refresher, check at the following tutorial[1].
- Use `numpy` to create and manipulate arrays. If you need a small introduction tutorial on numpy, check this link[2]. You must make use of trivial numpy vectorization[3] as indicated in the notebook.
- You can import `itertools` and use `itertools.combinations` to get the list of combinations.

### REQUIRED OUTPUT OF THIS SECTION:

You will create results similar to Figure 1. Show results of fitting two polynomials - $a = \{-2, -0.25, 1, 0.25\}$ and $a = \{1, 1\}$ - for four outlier ratios $r = 5\%$, $20\%$, $30\%$ and $70\%$. Display the synthesized circle in green, the best result of RANSAC in black, the inliers and outliers found by RANSAC in blue and red respectively.

[1] `https://docs.python.org/3/tutorial/`

[2] `https://numpy.org/doc/stable/user/quickstart.html`

[3] `https://www.pythonlikeyoumeanit.com/Module3_IntroducingNumpy/VectorizedOperations.html`

Show the distribution of the number of inliers found by RANSAC in a histogram (see Fig. 1). Note: the histogram does not correspond to the number of inliers found at each RANSAC iteration: RANSAC is run over $M$ iterations, and the final result counts as one entry in the histogram. Re-apply data generation and RANSAC 1000 times, and populate the histogram.

- Code that generates synthetic data points on a polynomial with different outlier ratios, applies RANSAC and exhaustive search, and plot the results as shown in Fig. 1
- Answer and discuss the following questions:
  - How many combinations (exhaustive search) exist for $N = 100$ points?
  - What about the number of RANSAC iterations with $r = 5\%$, 20%, 30% and 70%?
  - What about when $N = 100,000$ points?
  - Does exhaustive search on all the combinations of data points guarantee the optimal solution (in terms of number of inliers)? Why?
  - Does RANSAC always find close to the optimal number of inliers? Why? If not, would increasing the number of RANSAC iterations always improve that?
  - Discuss and compare the results obtained by RANSAC and exhaustive search in terms of number of inliers, speed, number of synthesized inliers, etc.
  - One of the challenges in using RANSAC for fitting polynomials is choosing appropriate values for the number of iterations and the inlier threshold. How would you go about selecting these values, and what factors should you consider when making this choice? Describe any tradeoffs that may exist between these two parameters, and provide an example scenario where a higher number of iterations or a higher inlier threshold might be preferable..

## 2. EXERCISE PART 2: IRLS AND NORMS FOR LINE FITTING

The second task of this assignment is to write a program to perform line fitting using the $L_1$ norm (employing IRLS and Linear Programming) and the $L_\infty$ norm (employing Linear Programming), as described in the lecture.

Write a program which takes as input a set of $N = 100$ 2D data points and fits the line by applying IRLS with $L_1$ norm and by applying Linear Programming with $L_1$ and $L_\infty$ norms.

To generate data points on a line, use the functions from Part 1.

We consider the line model $y = ax + b$ as a polynomial $\alpha = \{b, a\}$ and algebraic (vertical) cost.

You will create results similar to Figure 2. Show results of line fitting obtained for the outlier ratios $r = 0\%$, 1%, 10%, 50%, and 80%. In the same figure, plot the input data, the synthetic line in green, the result of your algorithm in black. Compare the results with RANSAC for $r = 80\%$.

A few hints about the implementation:

- use `linprog` from `SciPy` to perform Linear Programming (`from scipy.optimize import linprog`).
- test your solution with different lines, e.g. with negative slope

Required output of this section:

- Code that generates data points, runs IRLS with $L_1$ and LP with $L_1$ and $L_\infty$ norms. There is no need to include derivations of the LPs.
- Show results: plot the synthetic data points, the synthetic line, and the result of IRLS and LP, like in Figure 2
- Discuss the results obtained by these methods and compare and contrast them with RANSAC. What are the key differences between these two approaches? Under what circumstances might one be more suitable than the other?ompare with the results obtained with RANSAC.

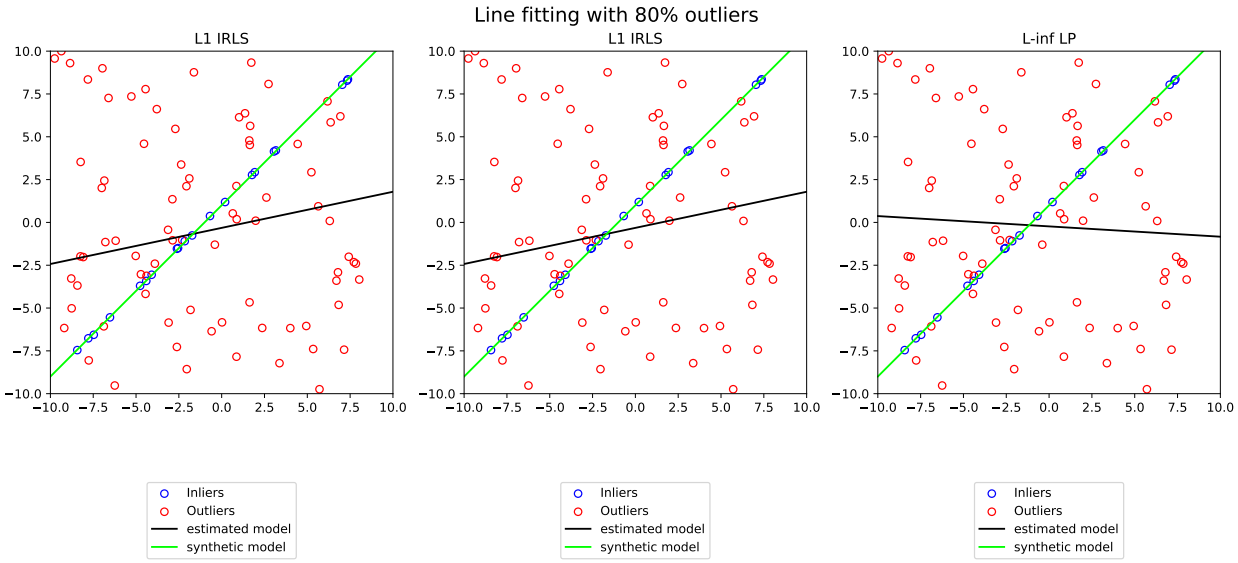


Figure 2. Line fitting with $r = 80\%$.