Tutorial 4 1/2: Workflow diagrams

In this tutorial, we'll review some key concepts on workflow diagrams and then show some solved examples.

Index

1. Brief introduction
   1.1. Why workflow diagrams?
   1.2. What do all workflow diagrams need to have?
2. Symbols recap
3. A little advice
4. Examples
   4.1. Image processing
   4.2. Clustering algorithm

---

1. Brief introduction

1.1. Why workflow diagrams?

They allow us to solve a problem without having to worry about all the syntactic difficulties of programming languages (e.g. "==" instaad of "="), while making the process comprehensible to others.

Moreover, when done right, they show that a person actually knows how to handle a programming task.

1.2. What do all workflow diagrams need to have?

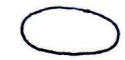
At a fundamental level, two things: **what** and **how**.

- What: Data, variables and features to use.
- How: Tools and methods (functions: self-made or third-party) to be applied on the what.

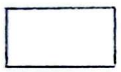
CamScanner

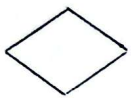
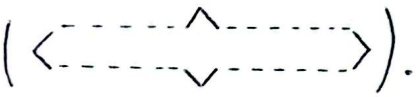
2. Symbols recap

We'll keep it simple.

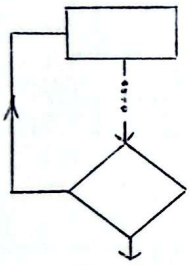
 : Start or end of the process (always present).

: Inputs/outputs of the process (also called data symbol).

 : Denotes an operation step.

 : A decision in the process flow ().

 : Indicates the flow direction between steps.

 : Usual structure of a loop

[more of a representation than a symbol].

3. A little advice

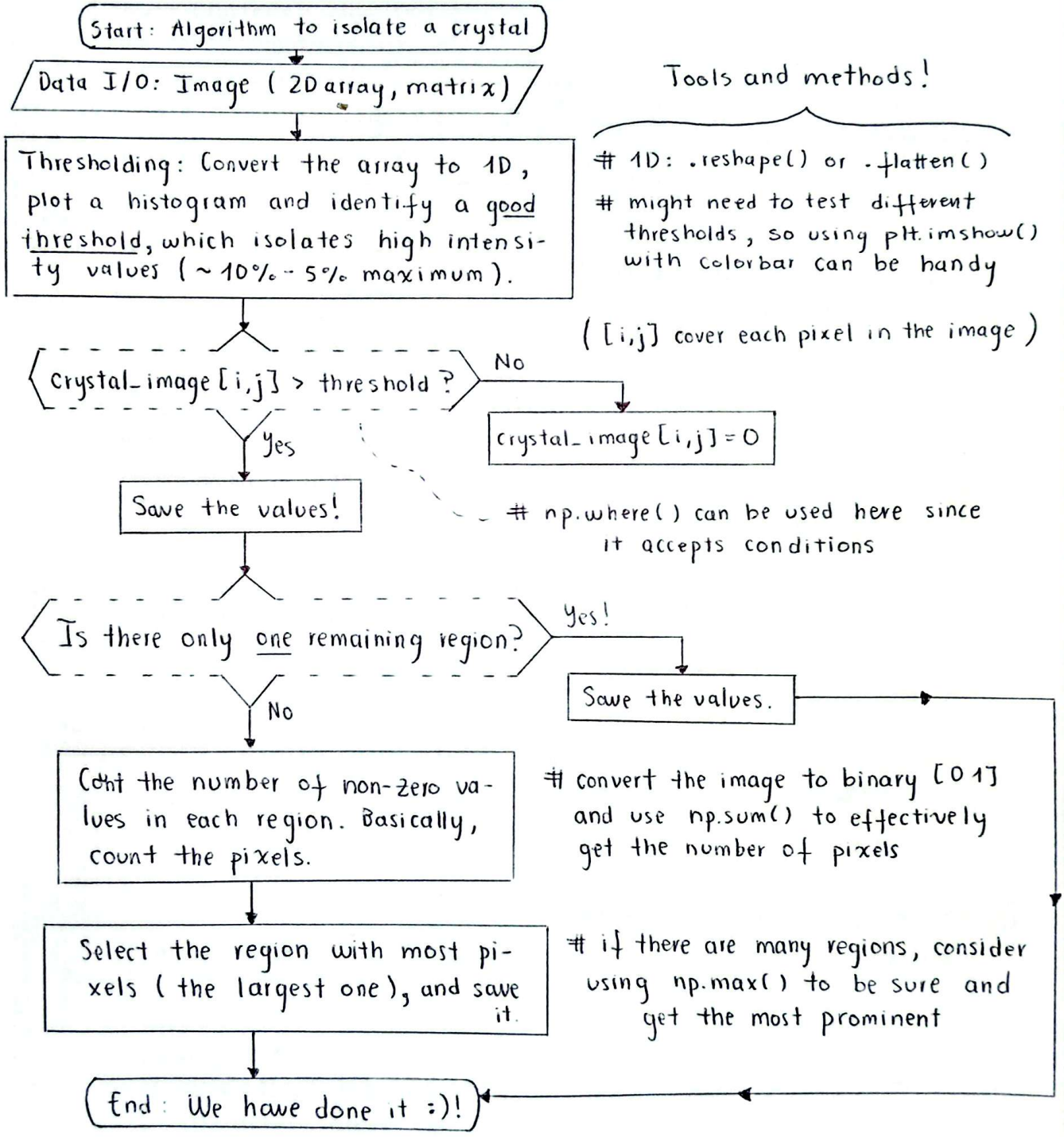
To start, all you need is an idea ☺. It does not have to be a great one, just one that, in principle, should work. I wouldn't recommend starting to write without having that. So take your time and don't rush :).

CamScanner

4. Examples

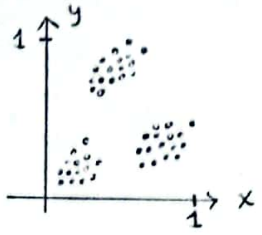
4.1. Image processing: Imagine you obtain a photograph of iron crystals (small scale), and you're asked to isolate the most prominent crystal from both the background and the rest of the image.

Comment: Britney Robalino Ramírez helped me with HER solution. Small things were added.



4.2. Clustering algorithm [from my Special-topics class]

Given the following distribution of points, how would you add a distinct label to each of the seemingly 3 groups? So that points with the same label belong to the same group.



Start: Clustering algorithm

1) Data I/O: Two arrays (x_i, y_i)
2) Number of clusters: 3

Tools, tricks and methods!

Get 3 initial random points in the specified domain: [0, 1] in x and y. [Centers]

can be done with np.random.rand(), which returns values between [0, 1]
of course, it should be done twice per point

Calculate the Euclidian distance of all the data points (one by one) to the 3 centers.

np.linalg.norm(•) can do this, and is remarkably fast
we would to specify the norm

Assign the closest center for each data point.

np.argmin() can help find the position of the minimum distance in the differences array

With these 3 clusters, recalculate better centers by getting the mean of the data points in each group.

# of course, np.mean()

Do centers still move / change?

Yes

define an epsilon (e.g. eps = 1e-6) to determine wheter centers still move

a while loop with a condition on the center's difference can be used

No!

Give a distinct label to each cluster and save results.

labels could be numbers [0 1 2]

End: We have labeled data! :)

Final comment: That'd be it. Sorry, I don't think any of the smiley faces looks pretty enough.

CamScanner