Practical advice for the labs
Miguel Á. Carreira-Perpiñán

1 Matlab programming

When programming machine learning algorithms (and other numerical algorithms) using Matlab:

- Program thinking of n dimensions, not 1 or 2. It is more general and usually easier.
- Many operations in machine learning algorithms involve vectors and matrices. Try to program these using vector and matrix operations in Matlab ("vectorized code") rather than loops, because 1) the code will be shorter and more readable (closer to the pseudocode and the mathematical formulas), 2) it will be faster (because Matlab is an interpreted language), and 3) you will save effort and avoid bugs. Example: a matrix-vector product `y = A*x` instead of a double loop $y_i = \sum_{j=1}^{n} a_{ij} x_j$ for $i = 1, \dots, n$.
- Machine learning algorithms can have a high time or space complexity, so to get a result in a few seconds you may need to run them on small datasets. You can do this by using toy examples that you create, or if using a real-world dataset (such as MNIST), by selecting a random sample from it. If your code takes more than a minute to run, it will make it slow to debug or use it with different parameter settings. Once your code is correct and if possible efficient, you can run it on bigger datasets.
- Machine learning algorithms often are randomized. Likewise, toy datasets are usually generated randomly. To make sure you can generate the exact dataset multiple times and run an algorithm and get the same result every time, fix the seed of the pseudorandom number generator. In Matlab: `rng(1778);` where 1778 is the seed. You can also save a toy dataset for later use.
- Matlab tips:
  - To suppress extra line feeds: `format compact`.
  - To get more decimals: `format long`.
  - To compare two matrices or vectors (by finding the largest difference): `max(abs(A(:)-B(:)))`.
  - To avoid distorted plots: `daspect([1 1 1])`.
  - To plot grayscale images with values in $[0, 1]$: `colormap(gray(256)); imagesc(I,[0 1]);`
    To plot images with negative and positive values: `colormap(parula(256)); imagesc(I);`
  - To solve a linear system `A*x = b`: `x = inv(A)*b` or, better still, `x = A \ b`.
  - To subtract a column vector `v` of $m \times 1$ from every column of a matrix `A` of $m \times n$: `A - repmat(v,1,n)` or, better still, `bsxfun(@minus,A,v)`.
- Matlab functions that are often useful:
  - Moments or other statistics: `sun mean std cov max min range mode`.
  - Array/matrix processing: `size length zeros ones sort sortrows find unique linspace cumsum reshape repmat squeeze bsxfun`.
  - Linear algebra: `diag trace linsolve inv eig eigs svd svds`.
  - Operators or symbols: `'  :  \ end NaN`.
  - Random numbers: `rand randi randn randperm`.
  - Plotting: `plot axis xlabel ylabel legend title text figure clf hold subplot drawnow scatter contour plot3 view image imagesc colormap colorbar gplot bar hist histc histogram`.
  - Others: `num2str pause ceil floor ind2sub sub2ind tic toc load save`.

2 Viewing animations

A good viewer for animated GIFs in Linux is `gifview`. It allows you to step through frames one by one and play/stop the animation.