Project 2

Fun with Polynomials

In this project, we will attempt to fit a polynomial to some (noisy) data and then use this approximation to find local maxima. Say we have some data points $y_i = f(x_i)$, $i = 1, ..., m$ and want to find a polynomial $p_n(x) = \sum_{i=0}^{n-1} a_i x^i$ meaning the coefficients a_i such that $p(x) \approx f(x)$.

This can be formulated as a least squares problem

$$\min_{a_i} \sum_{i=1}^{m} |p_n(x_i) - y_i|^2 . \tag{2.1}$$

Task 2.1

Reformulate (2.1) into the typical structure of a linear least-squares problem

$$\min_{\boldsymbol{a}} \|\boldsymbol{V}\boldsymbol{a} - \boldsymbol{y}\|_2^2 .$$

Identify the entries of $\boldsymbol{a}$, $\boldsymbol{y}$, and $\boldsymbol{V}$. What special structure does $\boldsymbol{V}$ have?

Task 2.2

Give the solution $\hat{\boldsymbol{a}} := \operatorname*{argmin}_{\boldsymbol{a}} \|\boldsymbol{V}\boldsymbol{a} - \boldsymbol{y}\|_2^2$.

Task 2.3

Given the QR-decomposition of $\boldsymbol{V} = \boldsymbol{Q}\boldsymbol{R}$, what is the resulting expression of $\hat{\boldsymbol{a}}$?

First we want to compare different approaches to calculate the solution to the least squares problem. To this end, we need implementations for various QR-decompositions.

PROGRAMMING TASK 2.4

Finish the implementations in `gs.m` and `mgs.m` for the classical and modified Gram-Schmidt (Algorithm 3 in the lecture) QR-decompositions. Use matrix-vector multiplications where possible. You can test for correctness by running `gs_test.m` and `mgs_test.m`. Similar test functions are available for most of the other programming tasks.

For the Householder QR-decomposition, we first need to implement a function which, given some input vector $\boldsymbol{a} \in \mathbb{C}^n$, returns the *unit-norm* Householder reflector $\boldsymbol{w}$ that, when applied to $\boldsymbol{a}$ zeroes out all but the first component of $\boldsymbol{a}$. That is

$$(\mathbf{I} - 2\boldsymbol{w}\boldsymbol{w}^{\mathrm{H}})\boldsymbol{a} = \begin{bmatrix} -\left\|\boldsymbol{a}\right\|_2 e^{j\phi_1} \\ \mathbf{0} \end{bmatrix}.$$

with the complex phase of a_1

$$e^{j\phi_1} = \frac{a_1}{|a_1|}.$$

PROGRAMMING TASK 2.5

Finish the implementation in `householder.m`. Make sure that the first element of $\boldsymbol{w}$ has the maximum possible absolute value as discussed in the lecture.

As we know from the lecture, we can use Householder reflections to triangularize a tall matrix $\boldsymbol{V}$. When using Householder reflections to calculate a QR-decomposition $\boldsymbol{V} = \boldsymbol{Q}\boldsymbol{R}$, we typically do not return the matrix $\boldsymbol{Q}$ explicitly, but the Householder vectors which can be used to construct $\boldsymbol{Q}$.

PROGRAMMING TASK 2.6

Implement the Householder QR decomposition in the file `hhqr.m`. Use the previously implemented `householder.m`. The decomposition should return the quadratic triangular matrix $\boldsymbol{R}$ and the matrix $\boldsymbol{W}$ containing the Householder vectors in the *lower triangular part*. Note that the test needs the implementation of `applyQHe.m` from the next task.

To apply the pseudo inverse $\boldsymbol{V}^{+} = \boldsymbol{R}^{-1}\boldsymbol{Q}^{\mathrm{H}}$ to a vector $\boldsymbol{a}$ we need two additional functions. One which applies the Householder transformations in $\boldsymbol{V}$ to a given vector and one which does the back substitution with the triangular matrix $\boldsymbol{R}$.

PROGRAMMING TASK 2.7

Finish the implementation of `applyQHe.m` which calculates $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{a}$ using the Householder vectors in $\boldsymbol{W}$.

PROGRAMMING TASK 2.8

Finish the implementation of `backsub.m` which calculates $\boldsymbol{R}^{-1}\boldsymbol{a}$ via back substitution. You can test for correctness with `backsub_test.m`.

Now we have implemented all of the functions required to try different approaches, to solve the least squares problem. To compare the approaches we try to fit to $y_i = \sin(1/x_i)$ where the x_i are linearly spaced between 0.1 and 1.

Use the file `plolyfit.m` for the following tasks.

PROGRAMMING TASK 2.9

For $n = 12$, calculate the optimal coefficients a_i using classical Gram-Schmidt, modified Gram-Schmidt, and the Householder decomposition.
Hint: Use the MATLAB function `vander` to generate $\boldsymbol{V}$.

PROGRAMMING TASK 2.10

Plot the polynomials with the coefficients that result from the different approaches. Compare with the original function $\sin(1/x)$. What do you observe?
Hint: The MATLAB function `y = polyval(a(end:-1:1),x)` can be used to evaluate the polynomials with the coefficients in `a` at the positions in `x`. Since MATLAB expects the coefficients in the reverse order compared to our definition we have to flip the vectors.

TASK 2.11

How does the result change for $n = 7$ and $n = 24$? Explain your observations.