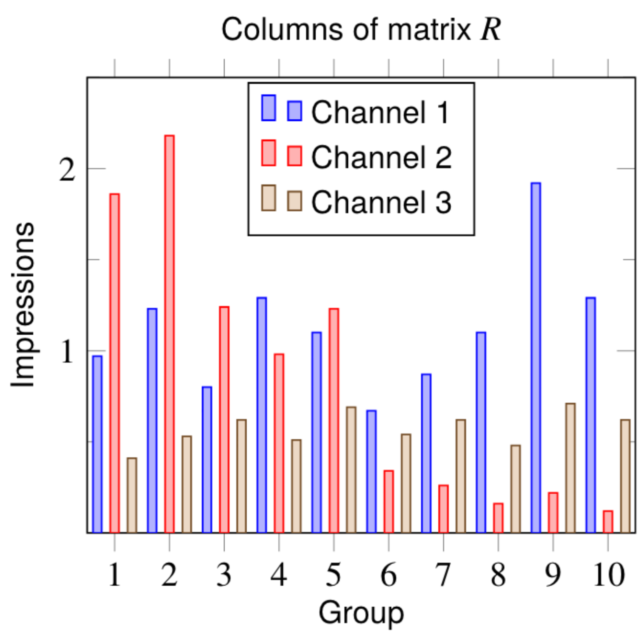
11 Least Squares



Unit 1: Vectors, Book ILA Ch. 1-5

Unit 2: Matrices, Book ILA Ch. 6-11 + Book IMC Ch. 2

Unit 3: Least Squares, Book ILA Ch. 12-14

- ***11 Least Squares***
- 12 Least Squares Data Fitting
- 13 Least Squares Classification

Outline: 11 Least Squares

- Least Square Problem
- Solution of Least Square Problem

- Examples

Survey Results

Videos summarizing some of the concepts of this class:

- https://www.3blue1brown.com/topics/linear-algebra

Running the code from the lectures by clicking on Binder:

- https://github.com/bioshape-lab/ece3

Final preparation:

- Exercises from HW and class
- Review session with exercises
- Mock exam

Least Squares Problem

Definition: Let be given a $m \times n$ matrix A and m-vector b. The least squares problem is the problem of choosing an n-vector x to minimize:

$$||Ax - b||^2.$$

- $||Ax - b||^2$ is called the objective function,
- If $\hat{x}$ is a solution of the linear equation $Ax = b$, then $\hat{x}$ is a solution of the least square problem. The converse is not true.
- $\hat{x}$ is a solution of least squares problem if $||A\hat{x} - b||^2 \leq ||Ax - b||^2$ for any other n-vector x.

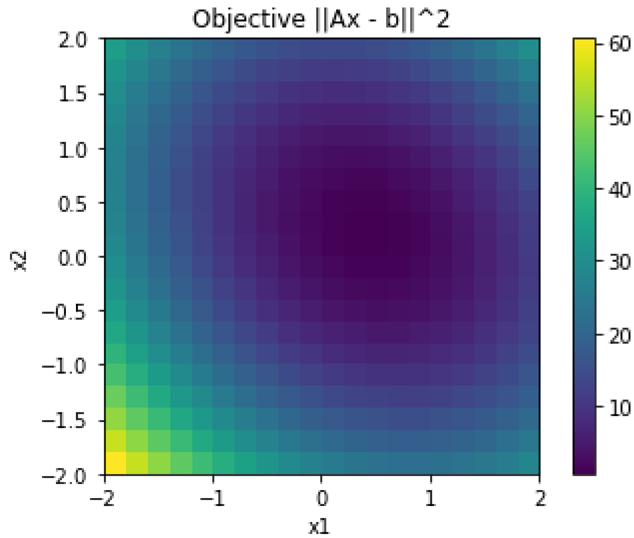
Exercise: Consider the matrix A and vector b as:

$$A = \begin{bmatrix} 2 & 0 \\ -1 & 1 \\ 0 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}.$$

Write the objective function associated to the least square problem defined by A and b in terms of entries of x.

In [8]:

```
# don't worry about this code
import numpy as np; import matplotlib.pyplot as plt
objective = lambda x : (2 * x[0] - 1) ** 2 + (- x[0] + x[1]) ** 2 + (2 * x[1] +
n_points, xmin, xmax, ymin, ymax = 20, -2, 2, -2, 2
x = np.arange(xmin, xmax, (xmax-xmin)/(n_points)); y = np.arange(ymin, ymax, (ym
for i in range(n_points):
    for j in range(n_points):
        Z[i, j] = objective(xx[i, j])
plt.imshow(Z, extent=[xmin, xmax, ymin, ymax]); plt.colorbar(); plt.xlabel("x1")
```



Outline: 11 Least Squares

- Least Square Problem
- **Solution of Least Square Problem**
- Examples

Least Square Solution

Proposition:

- Consider a least square problem $||Ax - b||^2$ for matrix A and vector b.
- Assume that A has linearly independent columns.

Then, there is a unique solution $\hat{x}$ to the least square problem, defined as:

$$\hat{x} = (A^T A)^{-1} A^T b = A^\dagger b.$$

- $A^\dagger = (A^T A)^{-1} A^T$ is called the pseudo-inverse of A.

Exercise (hard): Using the fact that:

$$||a + b||^2 = ||a||^2 + ||b||^2 + 2a^T b,$$

prove that $\hat{x}$ defined in the previous slide is indeed a solution.

- Hint: Show that for any other n-vector x, we have:

$$||Ax - b||^2 \geq ||A\hat{x} - b||^2.$$

- Hint 2: You will need to show that $A^T(A\hat{x} - b) = 0$.

In Python, we use:

- the function np.linalg.lstsq : returns the solution as the first element of the returned tuple
- the formula of the solution using transpose .T , inverse and matrix multiplication.

In [11]:
```
A = np.array([[2, 0], [-1, 1], [0, 2]]); b = np.array([1, 0, -1]); sol1 = np.lin
sol2 = np.linalg.lstsq(A, b); print(sol2)
```

```
[ 0.33333333 -0.33333333]
(array([ 0.33333333, -0.33333333]), array([0.66666667]), 2, array([2.44948974,
2.        ]))
/var/folders/dz/k1hb2xr94k558sjs416njdp40000gn/T/ipykernel_86527/614031835.py:2:
FutureWarning: `rcond` parameter will change to the default of machine precision
times ``max(M, N)`` where M and N are the input matrix dimensions.
To use the future default and silence this warning we advise to pass `rcond=None
`, to keep using the old, explicitly pass `rcond=-1`.
  sol2 = np.linalg.lstsq(A, b); print(sol2)
```

Outline: 11 Least Squares

- Least Square Problem
- Solution of Least Square Problem
- **Example**

Political Advertising

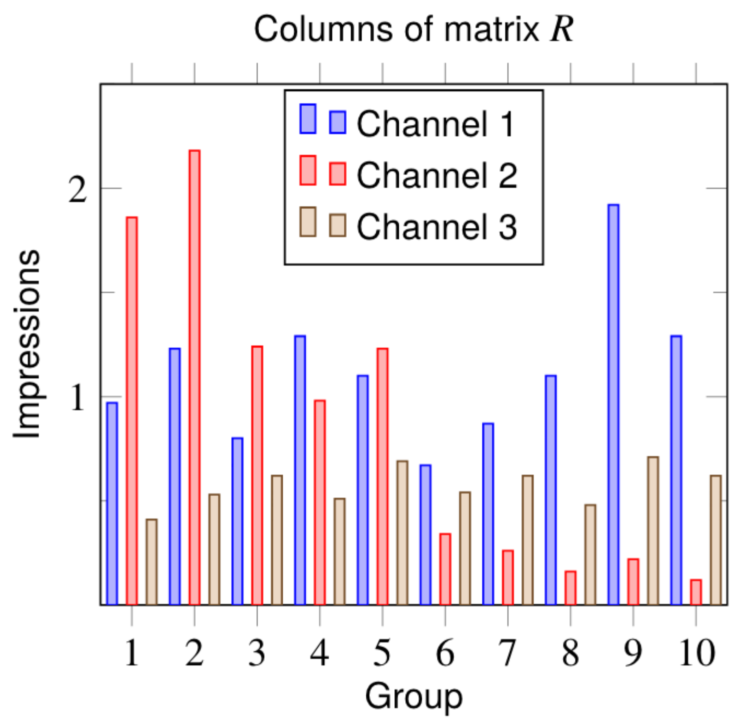
Example: Political Advertising

- A company wants to advertize to potential voters.
  - m demographics groups, n advertising channels
  - v^{target} is m-vector of target views ("impressions") per group
  - s is n-vector of spending per channel

    - R is $m \times n$ matrix of demographic reach of channels:
        - R_{ij} is number of views per dollar spent (in 1000/$)
- How much should be spent to be as close as possible to v^{target}?

Example: What is the optimal spending $\hat{s}$?

- $m = 10$ groups and $n = 3$ channels,
- $v^{target} = 1000.\, 1_m$.



In [4]:
```
import numpy as np

R = np.array([
    [0.97, 1.86, 0.41],
    [1.23, 2.18, 0.53],
    [0.8, 1.24, 0.62],
    [1.29, 0.98, 0.51],
    [1.1, 1.23, 0.69],
    [0.67, 0.34, 0.54],
    [0.87, 0.26, 0.62],
    [1.1, 0.16, 0.48],
    [1.92, 0.22, 0.71],
    [1.29, 0.12, 0.62]
])
```

Outline: 11 Least Squares

Resources: Book ILA Ch. 11

```
In [ ]:
```