Numerical Methods (CS 357)

Worksheet

Part 1. Objectives

- Understand Principal Component Analysis
- Be able to set up generalized Vandermonde matrices
- Be able to use Vandermonde matrices for interpolation and to take derivatives numerically

Part 2. Principal Components

The SVD is being used to for principal component analysis of a data set.
To that end, the SVD $\frac{1}{\sqrt{n-1}}Y = U\Sigma V^T$ of the mean-zero data matrix has been computed.
Where can the principal components be found?

(A) In the columns of U

(B) In the rows of U

(C) In the columns of V

(D) In the rows of V

Part 3. Vandermonde matrices

Let V be the generalized Vandermonde matrix for a set of functions $\varphi_1(x), \varphi_2(x), \varphi_3(x)$ at three points x_1, x_2, x_3.
Let V' be the generalized Vandermonde matrix for the functions $\varphi_1'(x), \varphi_2'(x), \varphi_3'(x)$ at the points x_1, x_2, x_3.
If the vector $y = [f(x_1), f(x_2), f(x_3)]$ contains function values of a function f to be interpolated, which of the following computes an approximation to point values of the derivative $[f'(x_1), f'(x_2), f'(x_3)]$?

(A) VV^{-1}

(B) $V'V^{-1}$

(C) $V'V^T$

(D) V^TV'

Part 4. Repetitions in Vandermonde matrices

Let V be the generalized Vandermonde matrix for a set of functions $\varphi_1(x), \ldots, \varphi_n(x)$ at points points $x_1, x_1, x_2 \ldots, x_{n-1}$. (Note that x_1 is repeated.)
What can you say about V?

(A) It cannot have full rank.

(B) It must be symmetric.

(C) Its first two columns are identical.

(D) It is orthogonal.

Part 5. Finding a cubic interpolant

In this problem, you will write code to find a cubic polynomial interpolant.
INPUT:

- `x`, a 4-vector of x coordinates
- `y`, a 4-vector of function values $f(x)$
- `xp`, a set of points where the interpolant should be plotted (see given initial code)

OUTPUT:

- `coeffs`, an array of coefficients $[c_0, c_1, c_2, c_3]$ so that $c_0 + c_1x + c_2x^2 + c_3x^3$ interpolates `y` at the points in the array `x`

```
import numpy as np
import numpy.linalg as la
import matplotlib.pyplot as pt

V = np.zeros((4,4))
V[:,0] =




coeffs =

pt.plot(xp, coeffs[0] + coeffs[1]*xp + coeffs[2]*xp**2 + coeffs[3]*xp**3)
pt.plot(x, y, "o")
```