k3math

Release 0.1.2

Zhang Yanpo

Nov 18, 2020

CONTENTS:

k3math is a toy math impl

CHAPTER

ONE

DOCUMENTATION FOR THE CODE

1.1 Classes

class k3math.**Vector**(**args*, ***kwargs*)

A `Vector` is a `list` supporting operations:

- +: vector adds vector
- −: vector subtracts vector
- *: vector times scalar
- **: vector powers scalar

inner_product(*b*)

Calculate inner product of two vector and returns a new Vector.

class k3math.**Matrix**(*vectors*)

determinant()

Calculate determinant of this matrix. E.g.:

```
| a b | = a*d - b*c
| c d |
```

Returns float

minor(*i*, *j*)

Make a new matrix without i-th row and j-th column.

solve(*ys*)

Solve equations:

```
|a00 a01 a02|   |x0|   |y0|
|a10 a11 a12| * |x1| = |y1|
|a20 a21 a22|   |x2|   |y2|
```

Parameters **y** (`Vector`) – a vector of *y0, y1, y2*.

Returns Vector

class k3math.**Polynomial**(*iterable=()*, */*)

It represents a polynomial: `y = a0 + a1 * x1 + a2 * x2 ...` Where *coefficients = [a_0, a_1, a_2 ..]*.

xs and ys is array of x-coordinate value and y-coordinate value. They are all real numbers.

xs = [1, 2, 3, 4, 5..] ys = [1, 2, 4, 7, 11..]

With xs and ys to calc the coefficients of a polinomial

degree is the highest power of polinomial: degree=2: y = a0 + a1*x + a2*x^2

classmethod fit (*xs*, *ys*, *degree*)

Find a polynomial curve with least squares method.

Parameters

- **x** (Vector) – Vector of x positions
- **y** (Vector) – Vector of y positions
- **degree** (*int*) – the highest power of variable *x* in the polynomial.

Returns Polynomial

classmethod get_fitting_equation (*xs*, *ys*, *degree*)

Curve fit with least squres

We looking for a curve:

Y = a0 + a1*x + a2*x^2

that minimize variance:

E = sum((Y[i]-ys[i])^2)

Partial derivatives about a0..an are:

E'a0 = sum(2 * (a0 + a1*xs[i] + a2*xs[i]^2 - ys[i]) * 1) E'a1 = sum(2 * (a0 + a1*xs[i] + a2*xs[i]^2 - ys[i]) * xs[i]) E'a2 = sum(2 * (a0 + a1*xs[i] + a2*xs[i]^2 - ys[i]) * xs[i]^2)

The best fit is a curve that minimizes E: or all partial derivatives are 0:

c00 c01 c02 | | a0 | | Y0 |
c10 c11 c12 | * | a1 | = | Y1 |
c20 c21 c22 | | a2 | | Y2 |

c00 = 2 * n c01 = 2 * sum(xs[i]) c02 = 2 * sum(xs[i]^2) Y0 = 2 * sum(ys[i])

c10 = 2 * sum(xs[i]) c11 = 2 * sum(xs[i]^2) c12 = 2 * sum(xs[i]^3) Y1 = 2 * sum(ys[i]*xs[i])

...

classmethod interpolation (*xs*, *ys*, *degree*, *x*)

guess value at x with polynomial regression

classmethod plot (*polynomials*, *rangex*, *rangey=None*, *width=120*, *height=20*, *points=()*)

Plot a polynomial with text:

```
poly = [3.5, 3.4, 1]

for l in Polynomial.plot([(poly, '.')],
                         rangex=[-1, 6],
                         width=40, height=10):
    print l
#                                 .
#                               ..
#                             ..
#                           ..
#                        ...
#                     ...
```

(continues on next page)

(continued from previous page)

```
#                          ...
#                      ....
#                  ....
#             .....
#  ...........
```

Parameters

- **polynomials** – list of a vector of polynomial coefficients and symbol:

```
[ ([1, 6], 'x'),     # y = 1 + 6x, plot with "x"
  ([2, 2, 2], '.'), # y = 2 + 2x + 2x^2, plot with "."
]
```

- **rangex** (*float*) – is a tuple of two floats that specifies range of x.
- **rangey** (*float*) – is a tuple of two floats that specifies range of y.
- **width** (*int*) – specifies plot graph width.
- **height** (*int*) – specifies plot graph height.
- **points** – other points to add to the plot. It is a vector of `(x, y[, char])`. `char` is optional to specify point mark. By default it is `X`.

Returns list of strings

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

k

INDEX

D

F

G

I

K

M

P

S

V