RLHS-Normal Equation

```python
t = np.array([2, 4, 6])
b = np.array([0.74, 0.79, 1])
A = np.vstack([np.cos(t), np.sin(t)]).T

n = A.T @ b
M = A.T @ A
print('n', repr(n))
print('M', repr(M))
```

```
n array([ 0.13584317, -0.20440937])
M array([[ 1.52235515, -0.15200858],
       [-0.15200858,  1.47764485]])
```

minimal norm $\|x\|_2$

```python
U = np.array([[-1, 0],
              [0, 1]])
S = np.array([2])
VT = np.array([[np.sqrt(3), 1],
               [-1, np.sqrt(3)]]) /2
b = np.array([7, 0])
temp1 = np.divide(np.dot(U.T, b), S)
weights_linear = np.dot(VT.T, temp1)
weights_linear
```

```
array([-3.03108891, -1.75      ])
```

拟合

```python
t = np.array([4, 0, -3.2,3.1])
y = np.array([9.3,0.9, -5, 7.1])
A = np.vstack([t, np.ones(len(t))]).T

m, b = la.lstsq(A, y)[0]
m,b
```

solu

```python
Sigma_p = 1 / Sigma.T
Sigma_p[Sigma_p == np.inf] = 0
x = VT.T @ Sigma_p @ U.T @ b
x
```

quad-fit

```python
t = np.array([-3.0, 1.8, -4.9])
y = np.array([3.3, 7.9, 15])
A = np.vstack([t, t**2]).T
coeffs = la.lstsq(A, y)[0]
coeffs
```

```
full_matrices = False.
a=0    sv= np.diag(s)@V
for i in range(k):
    a += np.outer(u[:,i],sv[i])
```

A: $m \times n$ rank=r

$r \times (m+n+1)$ size.

min(m,n)

$\|A\|_2 = \sigma_{max}$

$\|A-A_n\|_2 = \sigma_n$

$\|A^+\|_2 = \frac{1}{\sigma_{min}}$

[1,2,3]   1 2

1 2   1 2 3.

(4,1)

(4,1 |

$A = U\Sigma V^T$

The left SV AA^T ✓

Diagonal A^TA ✗

Diagonal AA^T ✓

The right A^TA ✓

The rank of $A = k$ ✓

$B = \sigma_1 u_1 v_1^T$ ✓

$A^+ = V\Sigma^+ U^T$

If A or, $A^{-1} = A^+$ ✓

If A^{-1} exist, $A^+ = A^{-1}$ ✓

Possible A^+ exist, when A^{-1} not ✓

For any mat. P-inv exist ✓

n^3, SVD, mm, LU