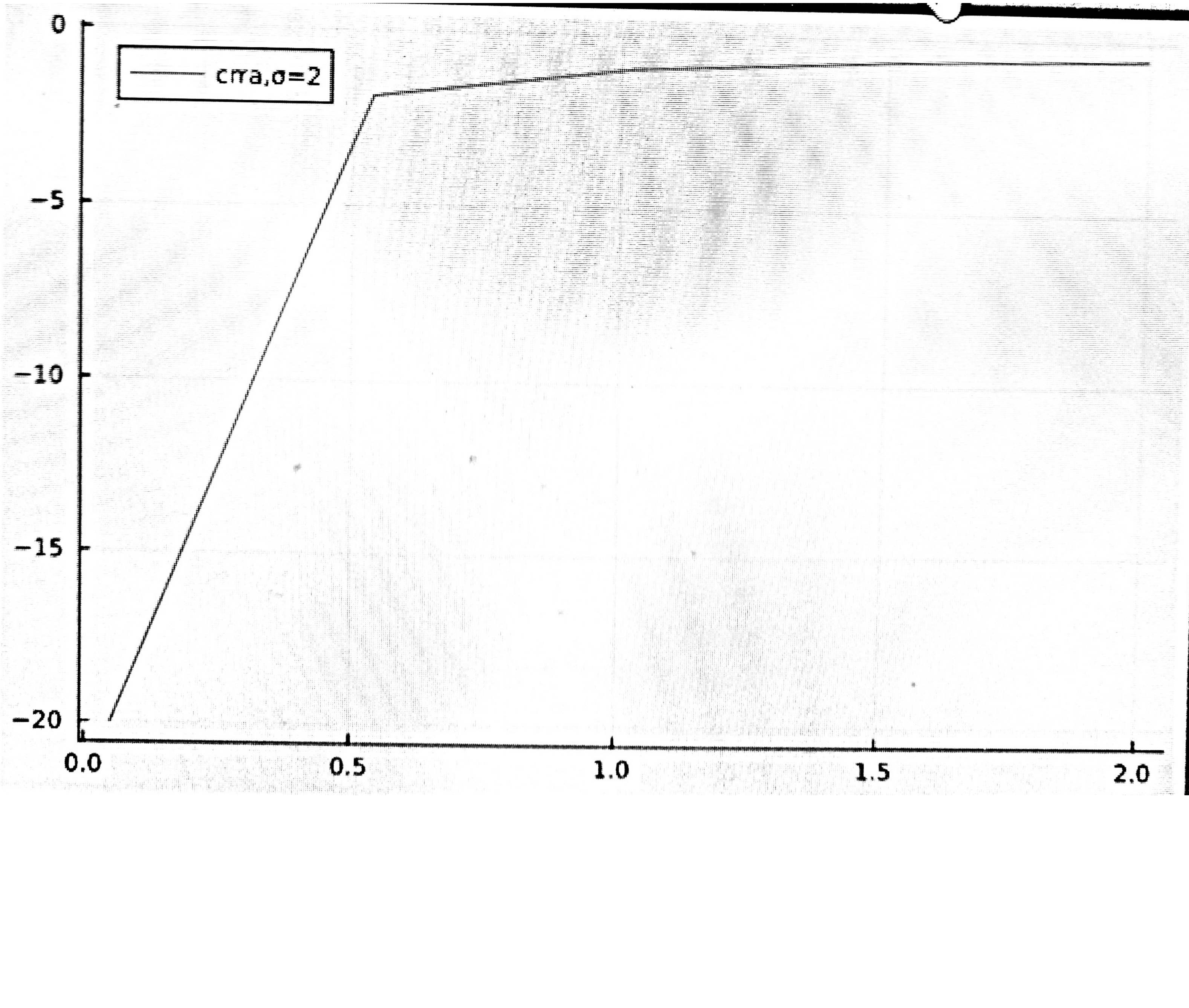
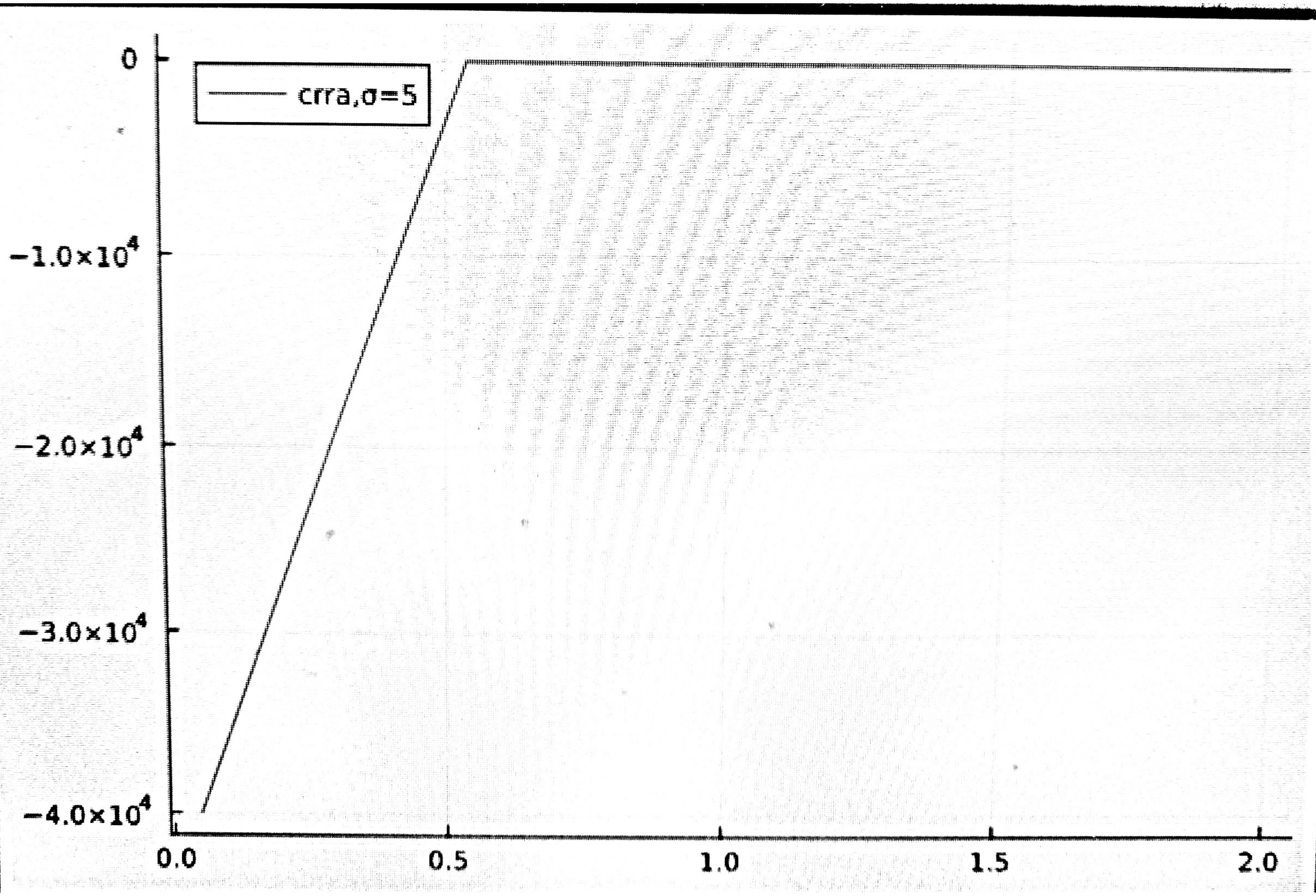
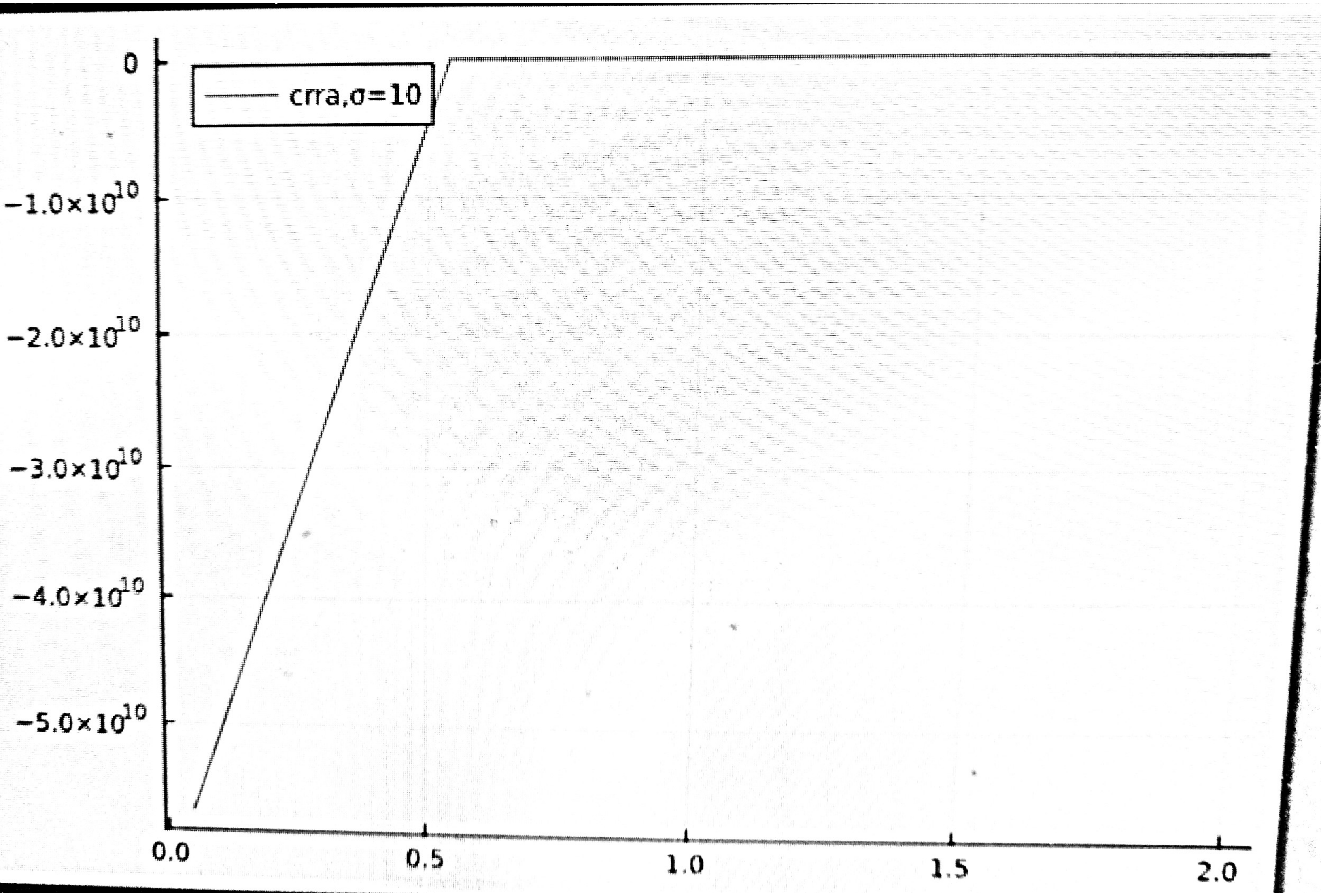
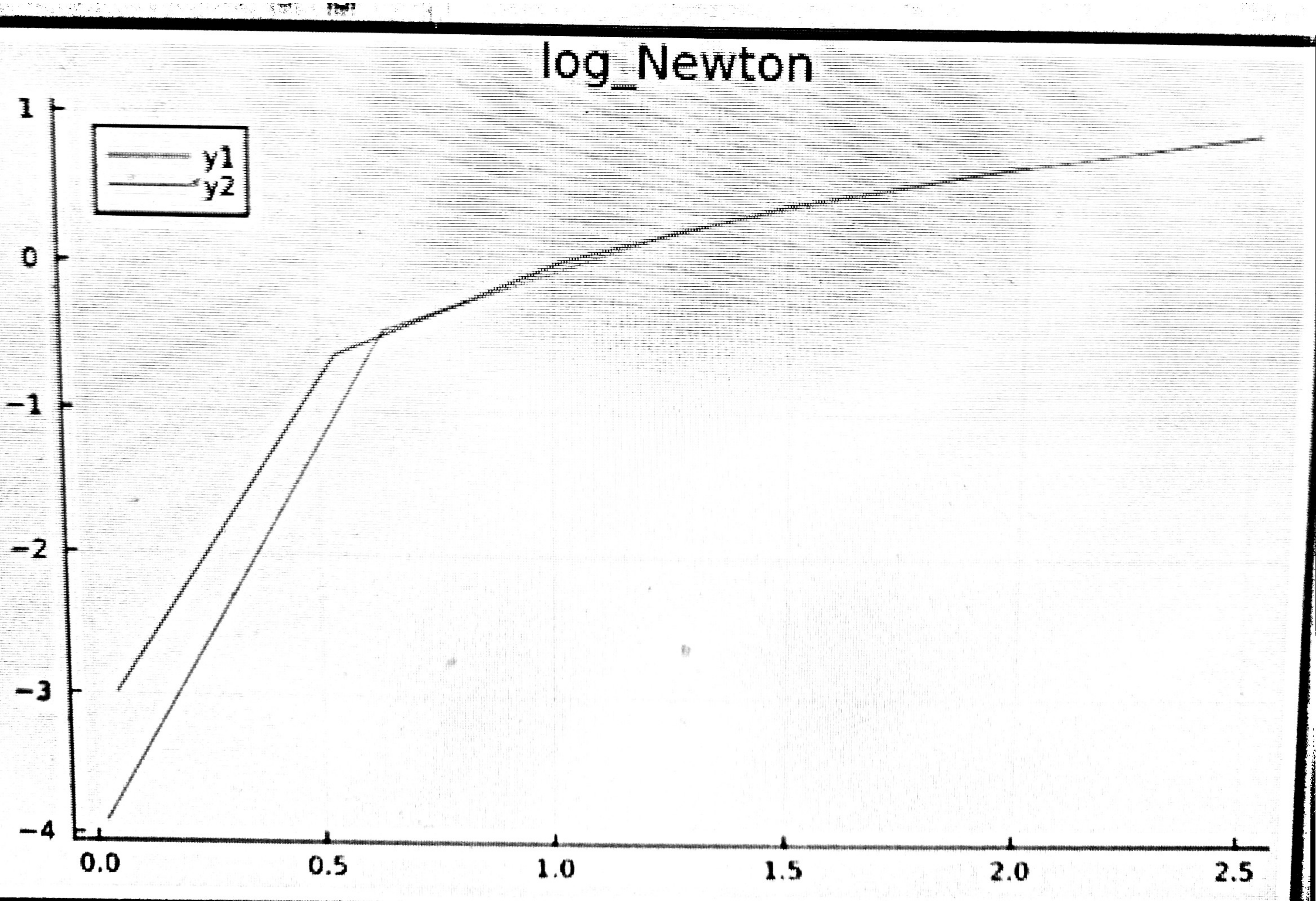

crra,σ=2
0
−5
−10
−15
−20
0.0
0.5
1.0
1.5
2.0



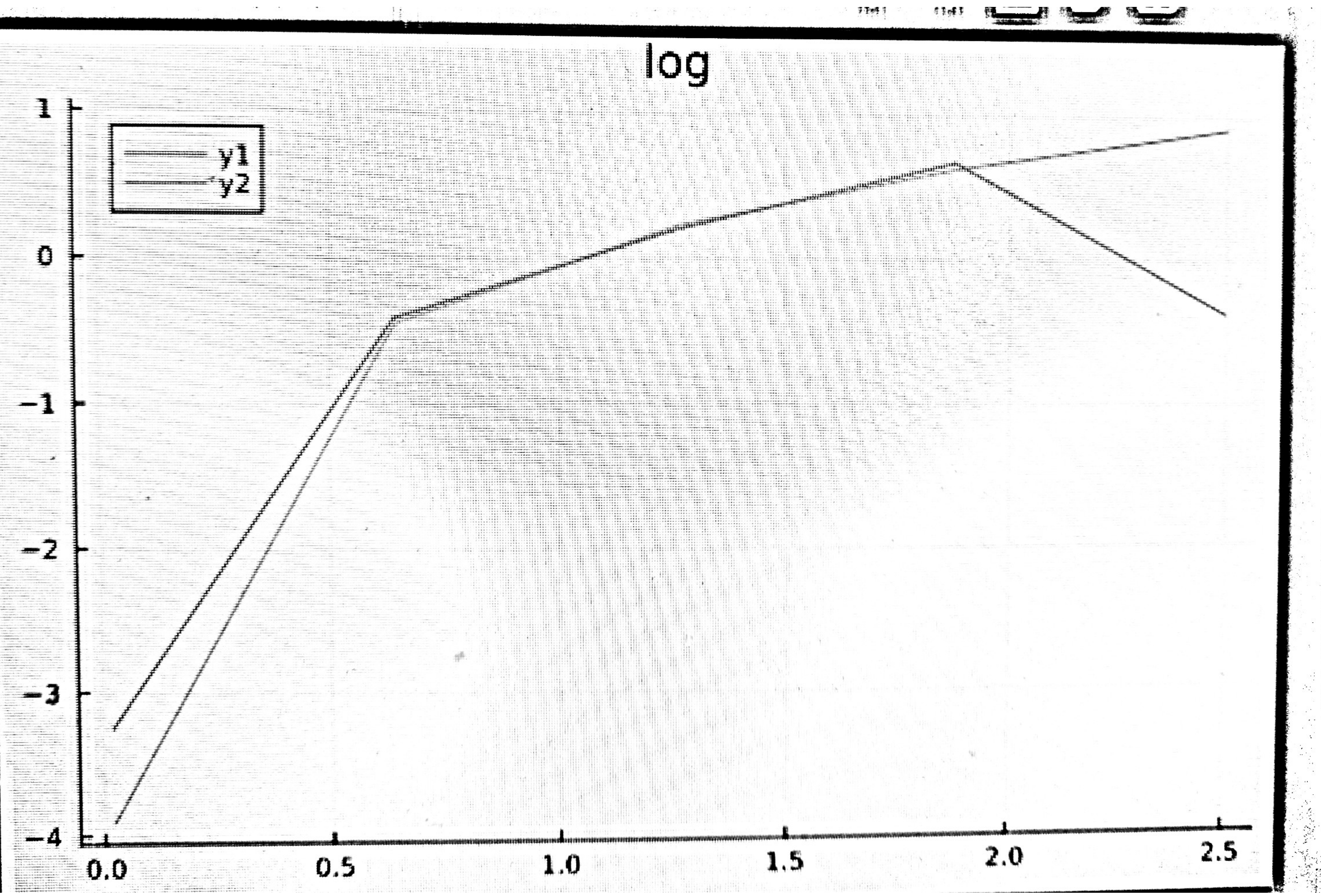
crra,σ=5
0
−1.0×10^4
−2.0×10^4
−3.0×10^4
−4.0×10^4
0.0
0.5
1.0
1.5
2.0

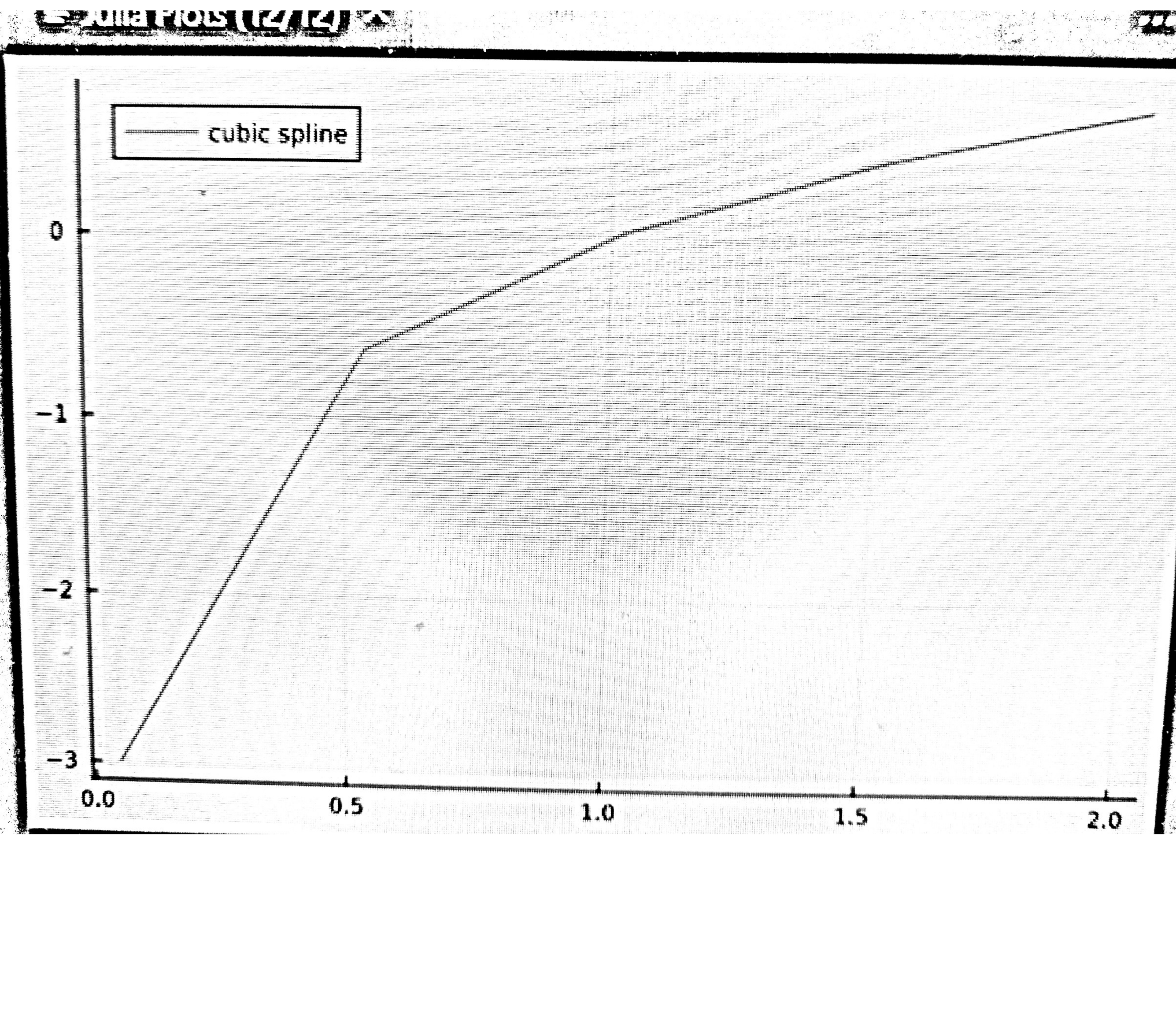


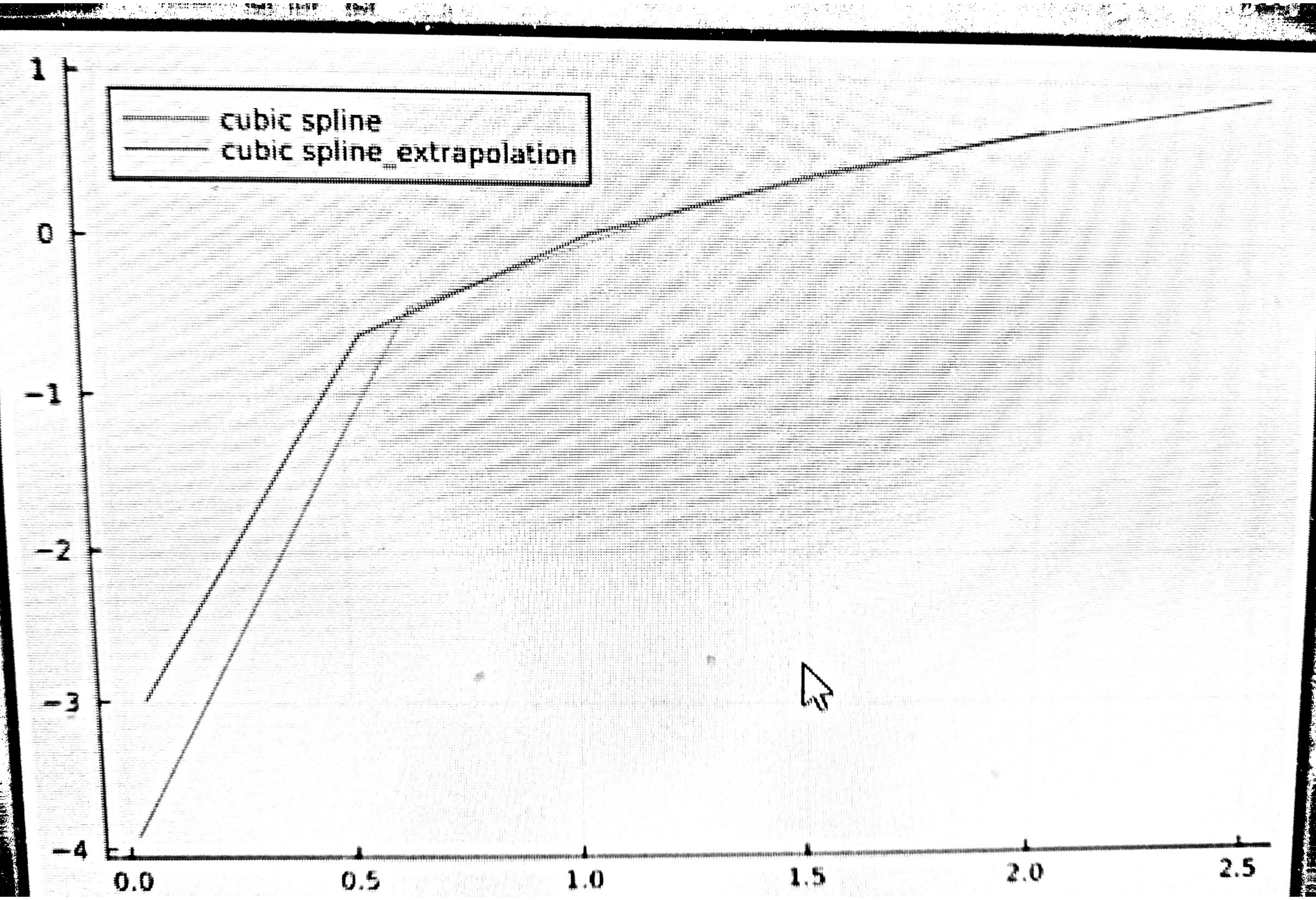
0
−1.0×10^10
−2.0×10^10
−3.0×10^10
−4.0×10^10
−5.0×10^10
0.0
0.5
1.0
1.5
2.0
crra,σ=10



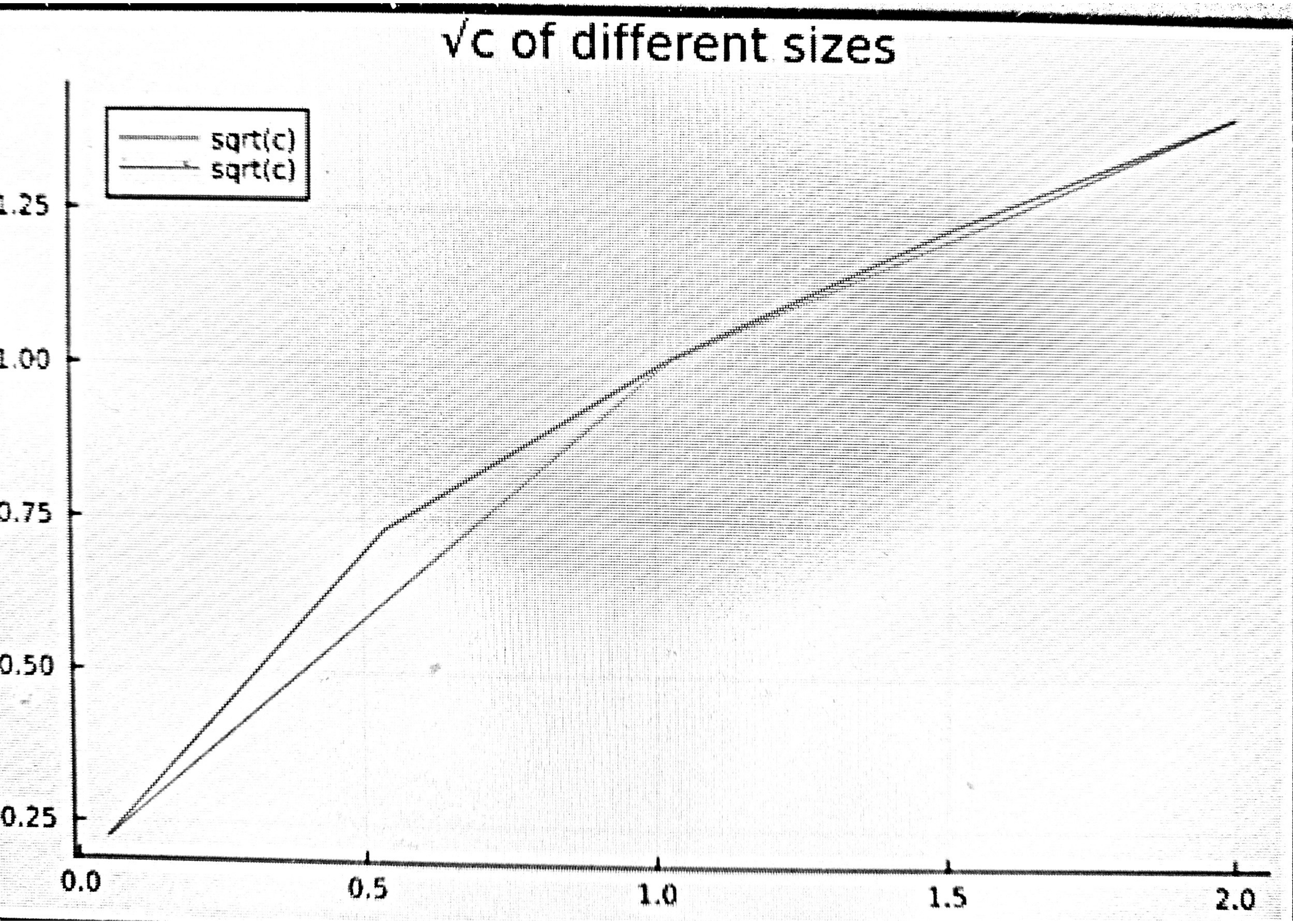
log_Newton
y1
y2
1
0
−1
−2
−3
−4
0.0
0.5
1.0
1.5
2.0
2.5


log
y1
y2
1
0
−1
−2
−3
−4
0.0
0.5
1.0
1.5
2.0
2.5

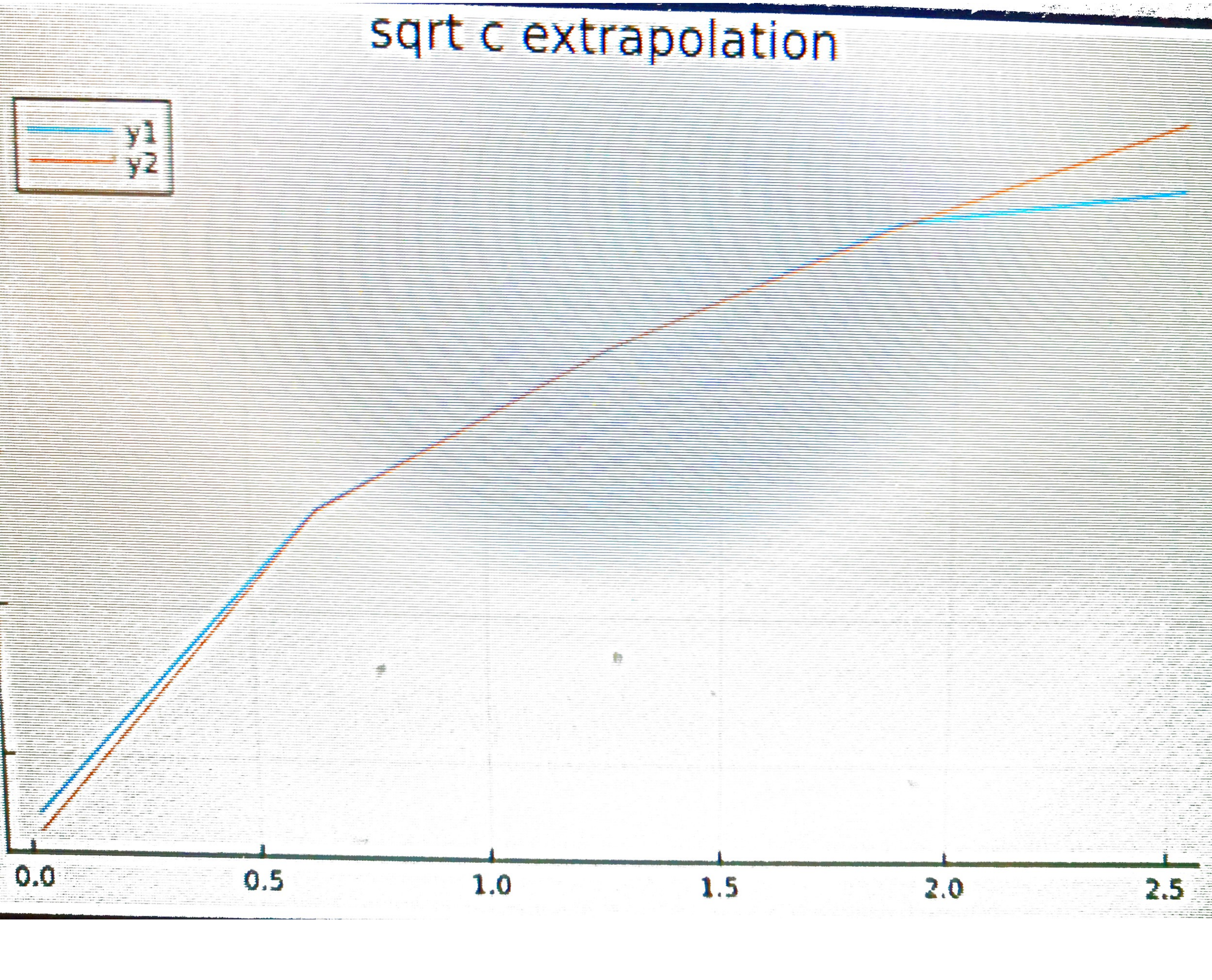

cubic spline
0
−1
−2
−3
0.0
0.5
1.0
1.5
2.0


cubic spline
cubic spline_extrapolation
1
0
−1
−2
−3
−4
0.0
0.5
1.0
1.5
2.0
2.5

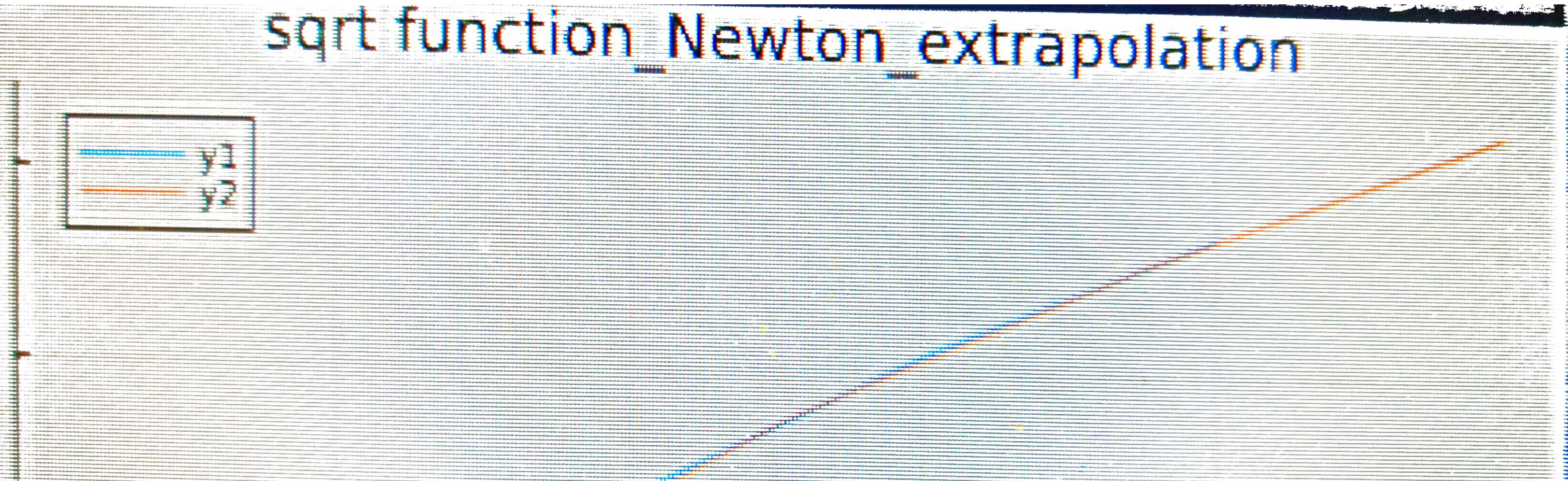

sqrt(c)
.25
.00
.75
.50
.25
0.0
0.5
1.0
1.5
2.0

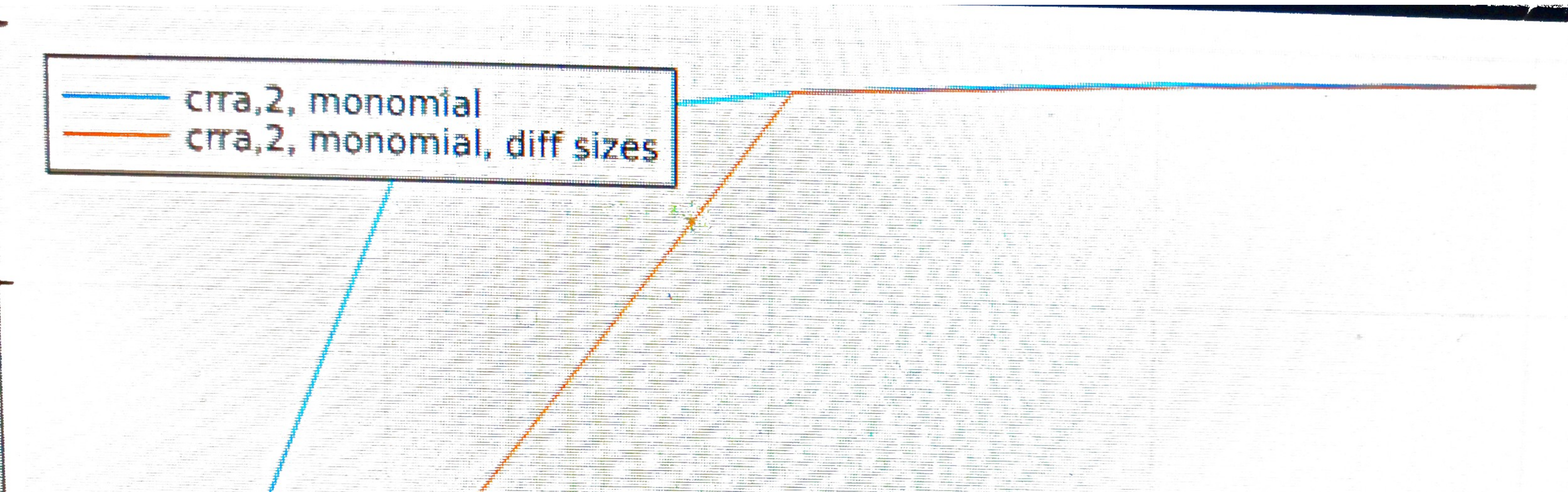


√c of different sizes
sqrt(c)
sqrt(c)
1.25
1.00
0.75
0.50
0.25
0.0
0.5
1.0
1.5
2.0

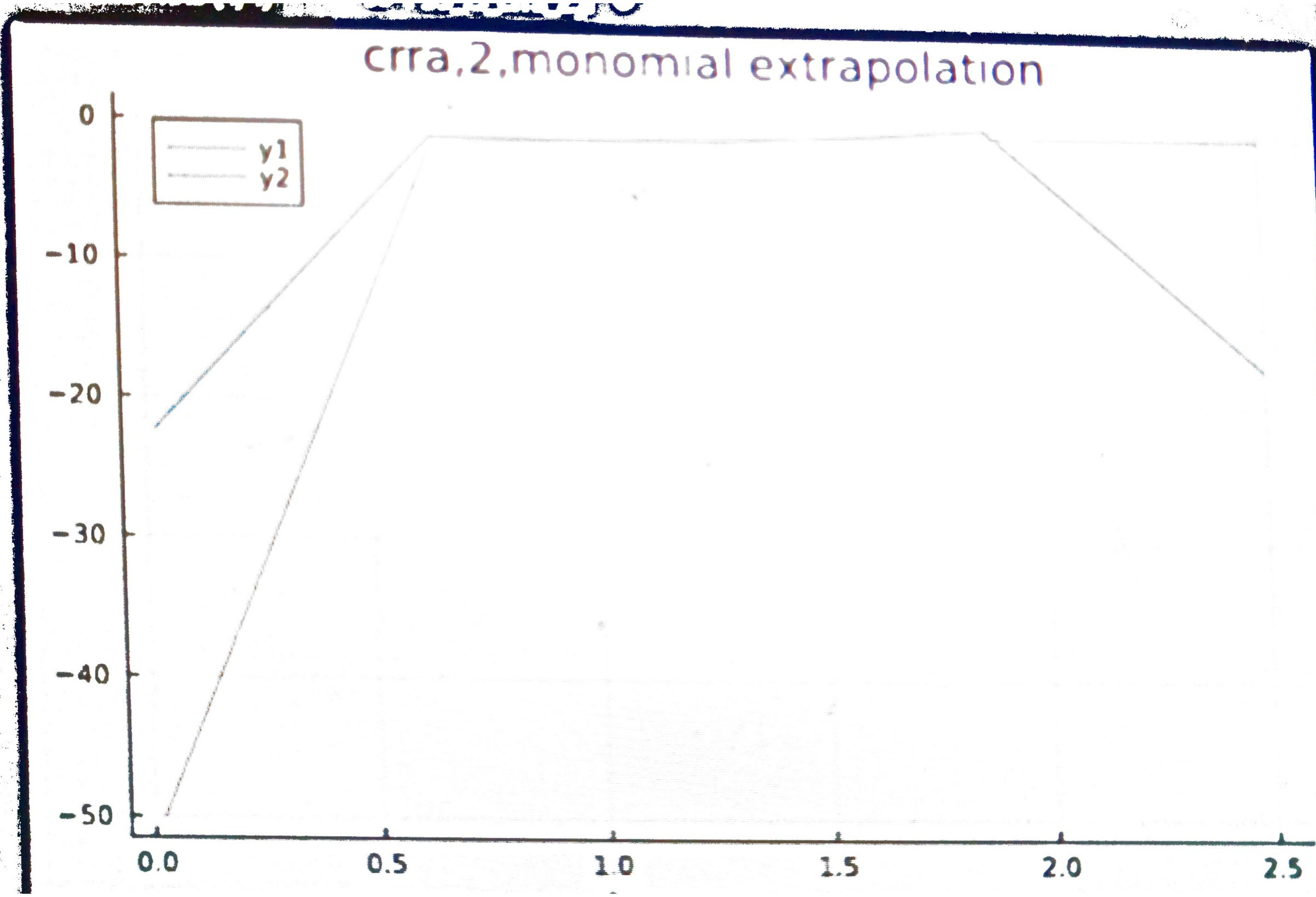


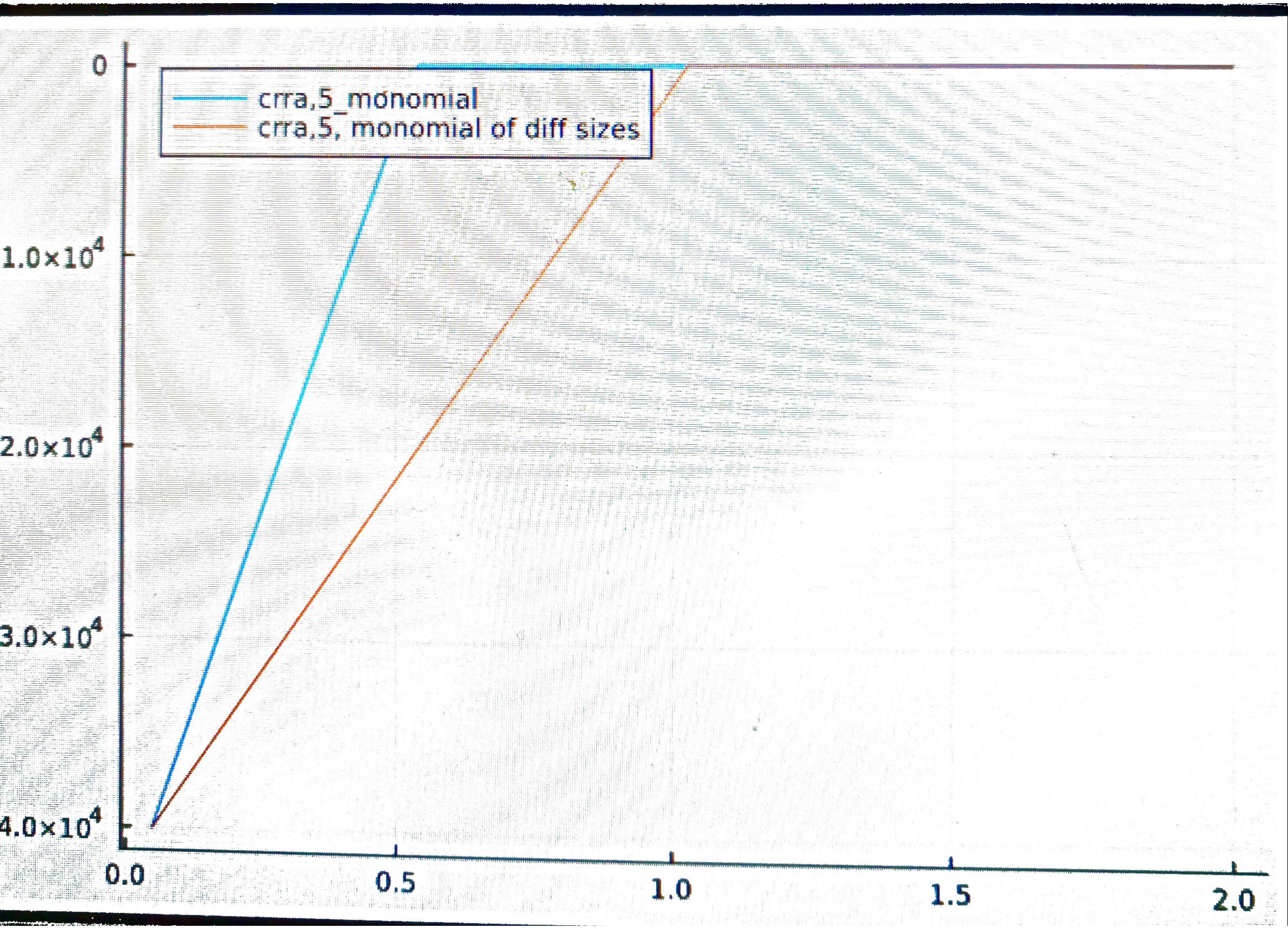
sqrt c extrapolation
y1
y2
0.0
0.5
1.0
1.5
2.0
2.5


sqrt function_Newton_extrapolation
y1
y2
0.0
0.5
1.0
1.5
2.0
2.5

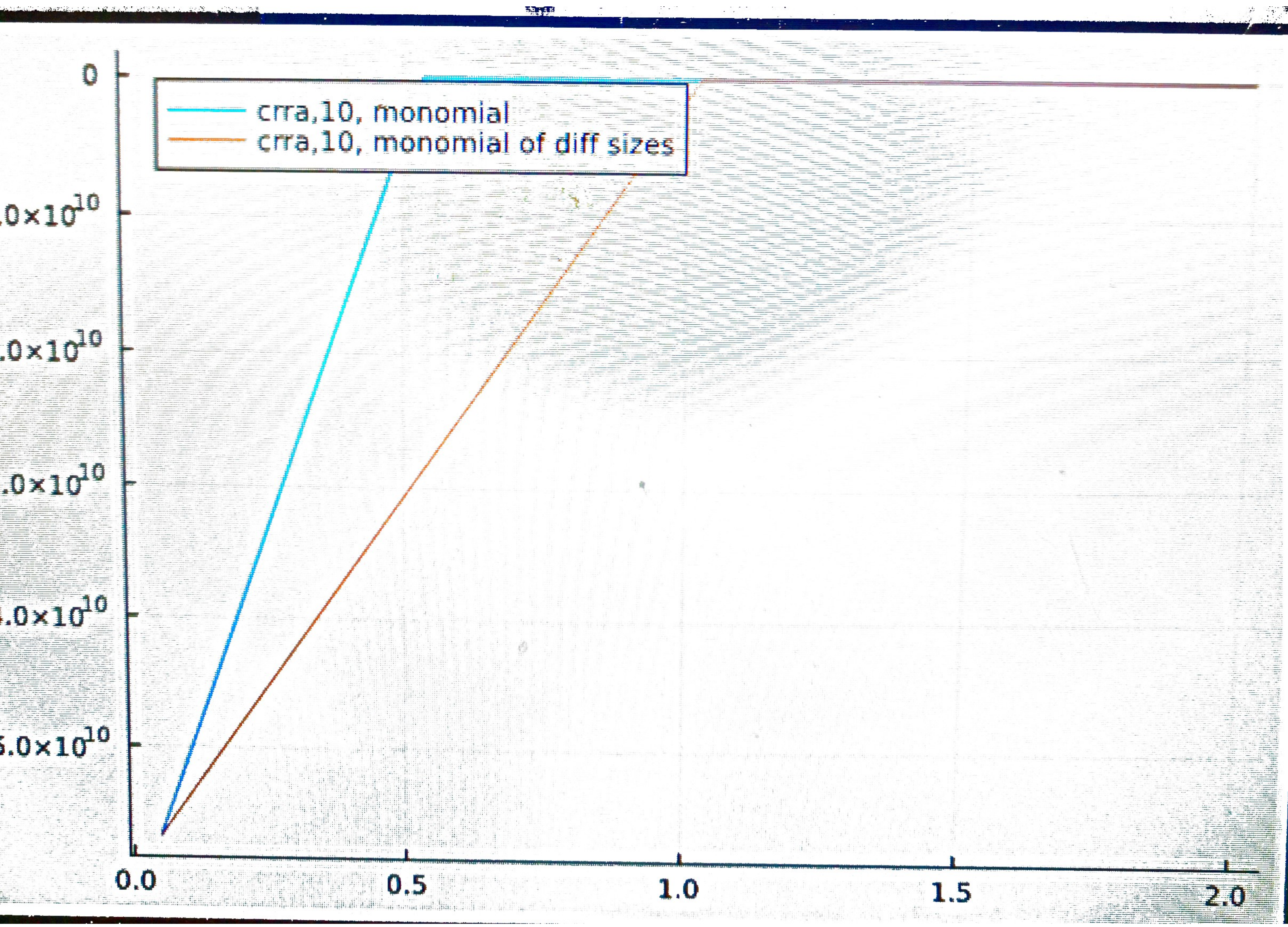


crra,2, monomial
crra,2, monomial, diff sizes
0.0
0.5
1.0
1.5
2.0


crra,2,monomial extrapolation
y1
y2
0
−10
−20
−30
−40
−50
0.0
0.5
1.0
1.5
2.0
2.5



crra,5_monomial
crra,5, monomial of diff sizes
0
1.0×10^4
2.0×10^4
3.0×10^4
4.0×10^4
0.0
0.5
1.0
1.5
2.0



0
.0×10^10
.0×10^10
.0×10^10
.0×10^10
5.0×10^10
0.0
0.5
1.0
1.5
2.0
crra,10, monomial
crra,10, monomial of diff sizes

Q1. a) I used polynomial interpolation with monomial and newton basis as well as cubic splines. For reference, I used the book by Ozteu.

Monomial basis.

I chose very simple grid, equally spaced, with only 5 points to start from.

polynomial fn $P_4(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4$

$$\begin{array}{cccccc} & x\downarrow & x^2\downarrow & x^3\downarrow & x^4\downarrow \\ \end{array}$$

$$\underbrace{\begin{bmatrix} 1 & 0.05 & 0.025 & 0.000125 & 6.25e\text{-}6 \\ 1 & 0.5375 & 0.288906 & 1.155284 & 0.0834668 \\ 1 & 1.025 & 1.05062 & 1.07689 & 1.10381 \\ 1 & 1.5125 & 2.28766 & 3.46008 & 5.23337 \\ 1 & 2.0 & 4.0 & 8.0 & 16.0 \end{bmatrix}}_{A\ \text{matrix}} \leftarrow \text{From Julia}$$

$$A * \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

y values are found for each function

We need to find vector of coefficients a?

$a = A \backslash y$ for each function

For $u(c) = \log c$ fn., the polynomial from Julia:

$p_n^1(x) = -3.436 + 9.29x - 10.06x^2 + 5.18x^3 - 0.978x^4$

For $u(c) = \sqrt{c}$, $p_n^2(x) = 0.15 + 1.583x - 1.19x^2 + 0.56x^3 - 0.103x^4$

For $u(c) = \frac{c^{1-\sigma}}{1-\sigma}$, $\sigma = 2$

$p_n^3(x) = -23.99 + 85.4x - 111.82x^2 + 61.5x^3 - 12x^4$

" — " $\sigma = 5$

$p_n^3(x) = -49177.5 + 196550x - 267417x^2 + 149719x^3 - 29500.75x^5$

— 4 $\sigma = 10$ (2)

$$p_n^3(x) = -6.99 + 2.796x - 3.8x^2 + 2.13x^3 - 4.197x^4$$

Figures are attached.

б) To assess accuracy of interpolation, <u>cross valida-tion method</u> was applied which is a leave-one-out resampling method that removes a single input point and uses the remaining points to predict the value at location of "hidden" point, and the predicted value is compared to the measured value. The "hidden" point is then added back to dataset, and a different point is "hidden" & so on. This process repeats through all input points. The error $MSE = \sqrt{\frac{\sum_1^n (\hat{z}(s_i) - z(s_i))^2}{n}}$ shall be $\rightarrow 0$.

To compare, I did the same for size of grid = 3 <u>(sorry, I made it manually as was getting errors when running loops)</u>

for $n=3$, I got for log $p_2^1(x) = -3.21 + 4.43x - 1.24x^2$

— " — TE $p_2^2(x) = 2.17 + 1.03x - 0.2x^2$

— " — CRDA
$\sigma = 2$ $p_2^3(x) = -21.5 + 29.9x - 9.76x^2$
$\sigma = 5$ $p_2^3(x) = -43129.5 + 63641.9x - 21038.6x^2$
$\sigma = 10$ $p_2^3(x) = -6.13 + 9.05x - 2.99x^2$

3

The differences (errors) for log fn. if size=5:

$$\begin{bmatrix} -0.0 \\ -6.66\ldots e{-16} \\ -2.16\ldots e{-15} \\ 6.66\ldots e{-16} \\ 1.11\ldots e{-16} \end{bmatrix}$$

" — " size = 3:

$$\begin{bmatrix} 0.0 \\ -8.26\ldots e{-16} \\ 1.11\ldots e{-16} \end{bmatrix}$$

The MSE for log when $n=5$ was $9.15\ldots e{-16}$

" — " $n=3$ was $4.13\ldots e{-16}$

Newton basis:

$$\pi_k(x) = \prod_{j=0}^{k-1} (x-x_j), \quad k=0,1\ldots n$$

$$p_n(x) = a_0 + a_1 \underbrace{(x-x_0)}_{\pi_1(x)} + a_2 \underbrace{(x-x_0)(x-x_1)}_{\pi_2(x)} + \ldots + a_n \underbrace{(x-x_0)\ldots(x-x_{n-1})}_{\pi_n(x)}$$

for size=5 grid above, f.ex.:

$$\pi_0(x) = \prod_{j=0}^{-1} (x-x_j) = 1$$

$$p_4(x) = a_0 + a_1(x-0.05) + a_2(x-0.05)(x-0.5375) + a_3(x-0.05)(x-0.5375)(x-1.025) + a_4(x-0.05)(x-0.5375)\cdot(x-1.025)(x-1.5125).$$

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0.49 & 0 & 0 & 0 \\ 1 & 0.98 & 0.48 & 0 & 0 \\ 1 & 1.46 & 1.43 & 0.6963 & 0 \\ 1 & 1.95 & 2.85 & 2.781 & 1.356 \end{bmatrix}}_{A} \cdot \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}}_{a} = \underbrace{\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}}_{y}.$$

manual

$$A = \begin{bmatrix} 1 & (x-x_1) & (x-x_1)(x-x_2) & \ldots \\ 1 & (x-x_2) & (x-x_2)(x-x_3) & \\ 1 & \vdots & & \vdots \\ 1 & \vdots & & \vdots \\ 1 & \vdots & & \vdots \end{bmatrix}$$

$\boxed{\text{Find coef } a:: A\backslash y.}$

I followed the code given in Ökten's book to do it using "diff" (divided differences) method.

diff vector for log fn. $n=5$ $\begin{bmatrix} -2.9957 \\ 4.87 \\ -3.64 \\ 2.12 \\ -0.979 \end{bmatrix}$ ← 0 difference ← from Julia

$p_n(x) = a_0 + a_1(x-x_0) + a_2(x-x_0)(x-x_1) + \dots a_n(x-x_0)\cdot(x-\dots(x-x_{n-1}))$

$a_0 = f(x_0)$, $a_1 = \dfrac{f(x_1)-f(x_0)}{x_1-x_0}$, $a_2 = \dfrac{\dfrac{f(x_2)-f(x_1)}{x_2-x_1} - \dfrac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2-x_0}$

to evaluate polynomial @ point x

$p_4(x) = -2.9957 + 4.87(\cancel{x-x_1}) - 3.64(x-x_1)(x-x_2) +$
$+ 2.12(x-x_1)(x-x_2)(x-x_3) - 0.979(x-x_1)(x-x_2)\cdot(x-x_3)(x-x_4)$

I used package in order to plot as there were errors in code for matrix A and Newton method

Cubic spline method (natural spline)

for $n=5$ data points
$S_i(x) = a_i + b_i x + c_i x^2 + d_1 x^3$
$i = 0,1,2,3,4$

Equations:

$$\begin{cases} S_1(x_1) = y_1 \\ S_2(x_2) = y_2 \\ S_3(x_3) = y_3 \\ S_4(x_3) = y_4 \\ S'_{i-1}(x_i) = S'_i(x_i) \\ S''_{i-1}(x_i) = S''_i(x_i) \\ S''_0(x_0) = S''_{n-1}(x_n) = 0 \end{cases}$$

If solving by hand, we get sy of equations:

$Ax = y$

$x = A \backslash y$

Julia's code used is taken from Orten's book.

all 3 methods were done for all functions (codes attached). all plots attached

b), c) accuracies were assessed & plotted for all fns.

3) Extrapolation over [0.02, 2.5] - done, $n=5$ (codes are attached) all plots attached.

cubic spline extrapolation for log fn is more accurate at the right end of grid. The same for Newton basis interpolation. For monomial basis both sides of extrapolation grid are deviating from fitted.

extrapolation for size=5, $\sqrt{x}$ fn, deviation more at right end. for monomial basis.

extrapolation for $\sqrt{x}$ fn. with Newton came the most accurate!

extrapolations for CRRA fn. are a way off at both sides.