Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа компьютерных технологий и информационных систем

**Отчёт по расчётному заданию №2b**

Дисциплина: Теория вероятности и математическая статистика

Выполнил студент гр. 5130901/20003 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Гусев М.М.

(подпись)

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Никитин К.В.

(подпись)

“ ” \_\_\_\_\_\_ 2024 г.

Санкт-Петербург

2024

Оглавление

[Исходные данные 3](#_Toc167837118)

[Формулы и табличные значения 4](#_Toc167837119)

[Определение характеристик измерений 5](#_Toc167837120)

[Последовательная полиномиальная аппроксимация 6](#_Toc167837121)

[Аппроксимация другими способами 15](#_Toc167837122)

[Вывод 20](#_Toc167837123)

[Приложение 21](#_Toc167837124)

# Исходные данные

nx = 41 – количество точек, в которых проводились измерения

ny = 10 – количество измерений в каждой точке

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| x | y1 | y2 | y3 | y4 | y5 | y6 | y7 | y8 | y9 | y10 |
| -2 | 55,5561 | 19,2686 | 26,2929 | 19,9065 | 27,5261 | 47,341 | -33,0894 | 13,72 | 35,3236 | 21,9196 |
| -1,9 | -19,8308 | -1,88102 | 45,4618 | 17,9815 | 63,1104 | -3,85713 | -8,04952 | 17,9153 | 28,5558 | 20,1347 |
| -1,8 | -7,85028 | -53,2914 | -7,63365 | 38,7993 | -42,0105 | -11,777 | 18,7123 | -13,5689 | -73,6905 | 27,1165 |
| -1,7 | -37,8004 | 11,5969 | -42,6847 | -4,94413 | 14,9078 | 1,84822 | 15,5582 | -34,8197 | 14,8925 | 27,1165 |
| -1,6 | 55,683 | -1,53277 | -5,21008 | 15,5406 | 19,1883 | 39,1093 | 34,2531 | -19,5659 | 5,47818 | -21,7328 |
| -1,5 | 8,78579 | 31,691 | 18,1855 | -10,7062 | 35,3726 | -7,91345 | 2,27799 | -34,0994 | 67,0968 | 2,30931 |
| -1,4 | -13,9104 | 20,3763 | -4,06166 | 32,1249 | 43,239 | 42,8988 | 5,91265 | 4,70017 | 25,4134 | 45,2858 |
| -1,3 | -24,7075 | -3,90157 | 5,97991 | 71,7425 | 66,1443 | 21,0057 | -2,06615 | -1,17994 | -28,3195 | 8,9738 |
| -1,2 | -25,8609 | -4,80842 | 32,5214 | -51,5519 | -24,1509 | 19,0216 | -11,8742 | 38,0467 | 3,13693 | -12,1835 |
| -1,1 | 32,4577 | -29,0356 | -23,7985 | -18,9447 | -1,8038 | 33,7451 | 23,9998 | 44,8503 | 13,9773 | 17,503 |
| -1 | -14,5867 | -27,8762 | -17,2865 | -36,051 | -51,9107 | 4,92459 | -34,2702 | -30,8905 | 9,89457 | 6,00533 |
| -0,9 | 17,1061 | -12,0743 | -14,5572 | -21,6452 | 19,1637 | -24,8965 | 12,8205 | 17,0799 | 44,1682 | 3,07291 |
| -0,8 | -49,7569 | 21,5107 | -40,0989 | -36,6974 | -72,1967 | -13,2032 | -16,0486 | -75,357 | 28,2358 | -40,0748 |
| -0,7 | 2,11157 | -44,0594 | -34,6221 | -7,84323 | -16,1633 | -37,1637 | -34,6474 | -78,6803 | -14,6641 | -39,9849 |
| -0,6 | -5,0658 | 17,8069 | -34,9983 | -83,3564 | -78,1036 | -56,0472 | -55,3746 | -25,764 | -98,8908 | 8,56759 |
| -0,5 | -16,9153 | -53,7687 | -34,9469 | -73,3201 | -69,2994 | -72,8752 | -57,2416 | -14,4457 | 10,8975 | -49,8361 |
| -0,4 | 7,90437 | -20,6477 | -60,1209 | -62,9717 | -81,8908 | -31,0109 | -7,10737 | -63,8919 | -76,0016 | 0,589628 |
| -0,3 | -56,2013 | -47,9037 | -47,0831 | -84,8878 | -19,1627 | -27,6572 | -15,9823 | -60,4558 | -55,9325 | -21,7978 |
| -0,2 | -0,80371 | -57,5786 | -7,97322 | -26,1241 | -11,6874 | -1,33681 | -17,0417 | -15,741 | 0,241642 | -39,2894 |
| -0,1 | -16,9677 | -22,5431 | -60,6463 | -22,0293 | -27,4863 | -45,0305 | 62,4567 | -50,7942 | -2,85731 | 21,4469 |
| 0 | -0,87484 | -6,41432 | -7,92234 | 27,7361 | 15,5396 | -20,8811 | 36,6115 | -9,79278 | -15,4399 | 8,31446 |
| 0,1 | -17,9939 | 4,36882 | 36,5608 | 17,5444 | 37,9238 | 50,7836 | 40,2244 | 55,0855 | 48,1354 | 59,9598 |
| 0,2 | 25,3264 | 45,1131 | 45,9897 | 14,9565 | 55,5708 | 69,4646 | 95,3541 | 44,1683 | 51,2475 | 50,2461 |
| 0,3 | 27,9426 | -0,29798 | 66,4292 | 75,5224 | 33,8217 | 48,5458 | 42,9369 | 52,5374 | 83,8296 | 52,0475 |
| 0,4 | 121,965 | 67,6767 | 54,658 | 81,4279 | 84,7257 | 118,077 | 129,066 | 86,7413 | 54,8034 | 22,5489 |
| 0,5 | 83,7792 | 118,952 | 85,1974 | 65,7751 | 22,4482 | 91,7766 | 62,3421 | 136,241 | 90,5583 | 84,6531 |
| 0,6 | 116,177 | 90,2455 | 121,251 | 58,0044 | 129,087 | 102,571 | 69,4904 | 165,986 | 106,91 | 100,596 |
| 0,7 | 62,473 | 93,4742 | 69,6246 | 91,9678 | 97,492 | 145,286 | 83,2676 | 77,3897 | 55,8068 | 88,2761 |
| 0,8 | 108,644 | 113,478 | 60,8322 | 48,5878 | 70,6321 | 80,257 | 65,3139 | 75,1203 | 62,5025 | 84,4399 |
| 0,9 | 97,099 | 64 | 82,2634 | 16,2106 | 70,9757 | 73,2072 | 44,8847 | 54,5947 | 71,8078 | 44,2113 |
| 1 | 3,4396 | 49,321 | 55,9221 | -12,8525 | 11,1579 | -2,22197 | -32,0647 | 13,2678 | 30,8 | 17,8447 |
| 1,1 | -7,74129 | -83,4796 | -43,5708 | -11,9447 | -36,5718 | 15,893 | 10,88 | -21,8318 | -18,9409 | -62,3596 |
| 1,2 | -92,497 | -86,3932 | -69,5324 | -41,7909 | -62,9659 | -84,0075 | -82,4729 | -119,186 | -50,1021 | -66,0127 |
| 1,3 | -174,071 | -106,526 | -110,5 | -93,2711 | -109,403 | -158,283 | -96,6519 | -100,401 | -103,136 | -98,9743 |
| 1,4 | -125,289 | -153,736 | -151,807 | -121,92 | -121,227 | -132,112 | -155,215 | -148,549 | -168,703 | -124,199 |
| 1,5 | -165,643 | -145,405 | -178,521 | -180,356 | -190,445 | -195,095 | -142,835 | -182,941 | -140,122 | -189,803 |
| 1,6 | -152,96 | -195,692 | -172,943 | -208,769 | -217,989 | -180,164 | -225,862 | -160,745 | -191,657 | -169,699 |
| 1,7 | -181,528 | -189,256 | -155,48 | -156,448 | -180,796 | -129,252 | -187,279 | -147,036 | -147,122 | -163,494 |
| 1,9 | -99,4935 | -136,136 | -143,633 | -103,095 | -101,439 | -78,8729 | -66,8386 | -98,7416 | -127,378 | -133,647 |
| 2 | -55,3619 | -25,8188 | -103,441 | -59,2556 | -109,945 | -34,4751 | -87,9867 | -66,6048 | -51,038 | -24,1991 |

Табл. 1 Результаты измерений

# Формулы и табличные значения

– среднее значение y для каждого xi

– несмещенная оценка дисперсии для каждого xi

Q = 0,9 – доверительная вероятность

Границы доверительного интервала мат. ожидания:

– нижняя граница доверительного интервала мат. ожидания, где  
 – коэффициент Стьюдента

Параметрический толерантный

– нижний параметрический толерантный предел

– нижний параметрический толерантный предел, где  
– толерантный множитель

– критерий Кохрена

– табличное критическое значение критерия Кохрена для 40 выборок по 10 измерений

# Определение характеристик измерений

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| x |  |  |  |  | Нижняя граница доверительного интервала | Верхняя граница доверительного интервала |
| -2 | 23,37 | 565,6 | -36,91 | 83,66 | 15,85 | 30,89 |
| -1,9 | 15,95 | 649,5 | -48,65 | 80,56 | 7,894 | 24,01 |
| -1,8 | -12,51 | 1277 | -103,1 | 78,10 | -23,82 | -1,214 |
| -1,7 | -3,304 | 667,5 | -68,80 | 62,19 | -11,47 | 4,864 |
| -1,6 | 12,12 | 650,9 | -52,55 | 76,79 | 4,052 | 20,18 |
| -1,5 | 11,29 | 804,3 | -60,59 | 83,19 | 2,331 | 20,26 |
| -1,4 | 20,19 | 449,0 | -33,51 | 73,91 | 13,49 | 26,89 |
| -1,3 | 11,36 | 1135 | -74,06 | 96,79 | 0,7104 | 22,02 |
| -1,2 | -3,770 | 772,9 | -74,25 | 66,70 | -12,56 | 5,021 |
| -1,1 | 9,291 | 687,6 | -57,18 | 75,77 | 1,002 | 17,58 |
| -1 | -19,20 | 431,3 | -71,85 | 33,44 | -25,77 | -12,63 |
| -0,9 | 4,023 | 484,4 | -51,77 | 59,82 | -2,936 | 10,98 |
| -0,8 | -29,36 | 1221 | -117,9 | 59,23 | -40,42 | -18,31 |
| -0,7 | -30,57 | 522,9 | -88,54 | 27,40 | -37,80 | -23,33 |
| -0,6 | -41,12 | 1603 | -142,6 | 60,38 | -53,78 | -28,46 |
| -0,5 | -43,17 | 814,9 | -115,5 | 29,19 | -52,20 | -34,14 |
| -0,4 | -39,51 | 1114 | -124,1 | 45,11 | -50,07 | -28,95 |
| -0,3 | -43,70 | 491,0 | -99,88 | 12,46 | -50,71 | -36,69 |
| -0,2 | -17,73 | 348,6 | -65,06 | 29,59 | -23,63 | -11,82 |
| -0,1 | -16,44 | 1331 | -108,9 | 76,07 | -27,98 | -4,904 |
| 0 | 2,68 | 358,1 | -45,28 | 50,65 | -3,296 | 8,671 |
| 0,1 | 33,25 | 611,9 | -29,44 | 95,96 | 25,43 | 41,08 |
| 0,2 | 49,74 | 486,5 | -6,171 | 105,6 | 42,76 | 56,71 |
| 0,3 | 48,33 | 597,6 | -13,64 | 110,3 | 40,60 | 56,06 |
| 0,4 | 82,16 | 1152 | -3,883 | 168,2 | 71,43 | 92,90 |
| 0,5 | 84,17 | 960,6 | 5,602 | 162,7 | 74,37 | 93,97 |
| 0,6 | 106,03 | 932,6 | 28,61 | 183,4 | 96,37 | 115,6 |
| 0,7 | 86,50 | 617,3 | 23,52 | 149,4 | 78,64 | 94,36 |
| 0,8 | 76,98 | 428,6 | 24,49 | 129,4 | 70,43 | 83,52 |
| 0,9 | 61,92 | 523,3 | 3,935 | 119,9 | 54,69 | 69,15 |
| 1 | 13,46 | 725,0 | -54,79 | 81,72 | 4,94 | 21,97 |
| 1,1 | -25,96 | 973,3 | -105,0 | 53,11 | -35,83 | -16,10 |
| 1,2 | -75,49 | 500,1 | -132,1 | -18,80 | -82,56 | -68,42 |
| 1,3 | -115,1 | 767,2 | -185,3 | -44,90 | -123,8 | -106,3 |
| 1,4 | -140,2 | 295,9 | -183,8 | -96,66 | -145,7 | -134,8 |
| 1,5 | -171,1 | 447,7 | -224,7 | -117,4 | -177,8 | -164,4 |
| 1,6 | -187,6 | 603,2 | -249,9 | -125,3 | -195,4 | -179,8 |
| 1,7 | -163,7 | 408,7 | -215,0 | -112,5 | -170,1 | -157,3 |
| 1,8 | -168,7 | 558,0 | -228,6 | -108,8 | -176,1 | -161,2 |
| 1,9 | -108,9 | 650,4 | -173,5 | -44,27 | -116,9 | -100,8 |
| 2 | -61,81 | 931,0 | -139,1 | 15,53 | -71,46 | -52,16 |

Табл. 2 Характеристики измерений

Критерий Кохрена – , меньше критического значений => измерения равноточные.

Т.к. измерения равноточные, многократные, характеристики погрешностей измерений неизвестны, то используется МНК, средняя оценка дисперсии вычисляется по формуле

,

# Последовательная полиномиальная аппроксимация

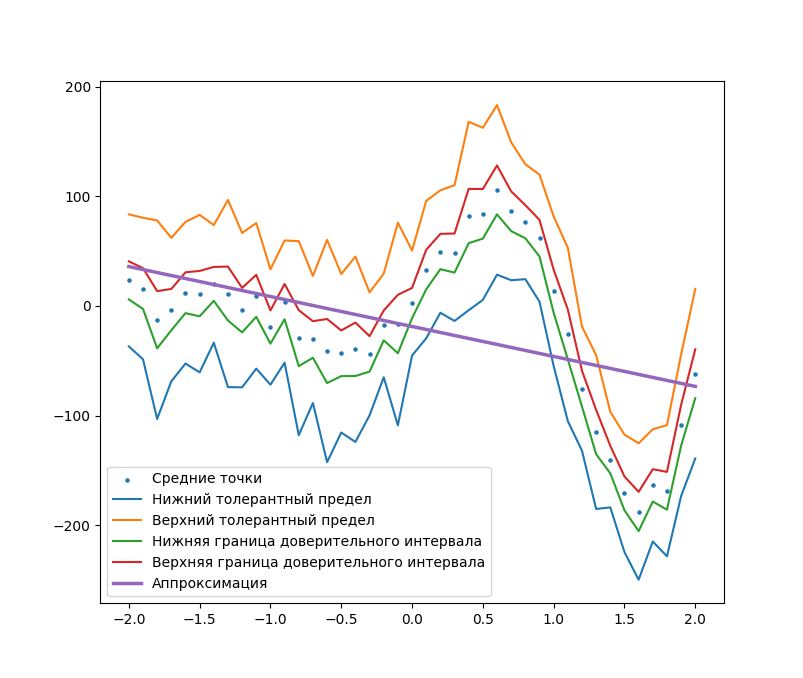
**Степень полинома *q* = 0**

Изображение выглядит как текст, диаграмма, линия, График

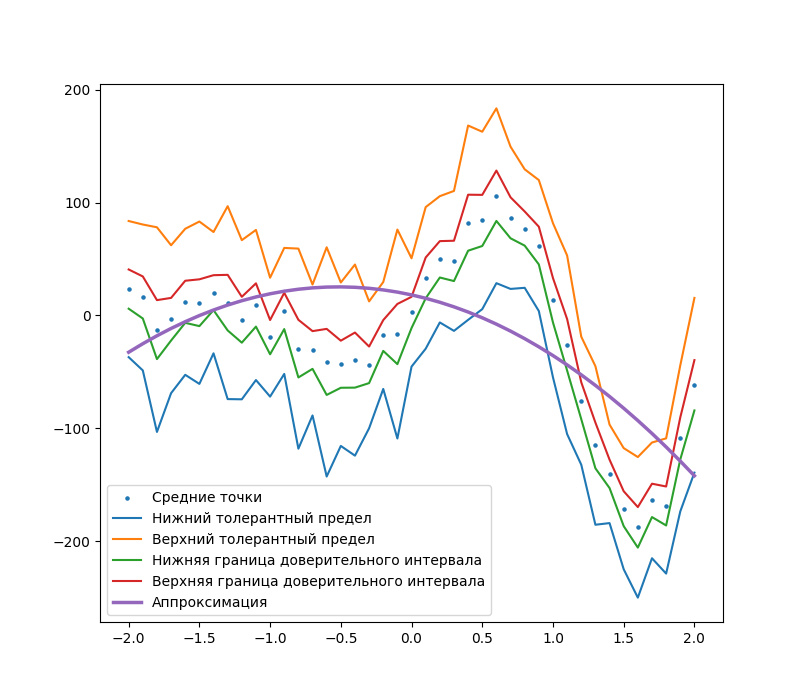
Автоматически созданное описание

Рис. 1 Степень *q* = 0

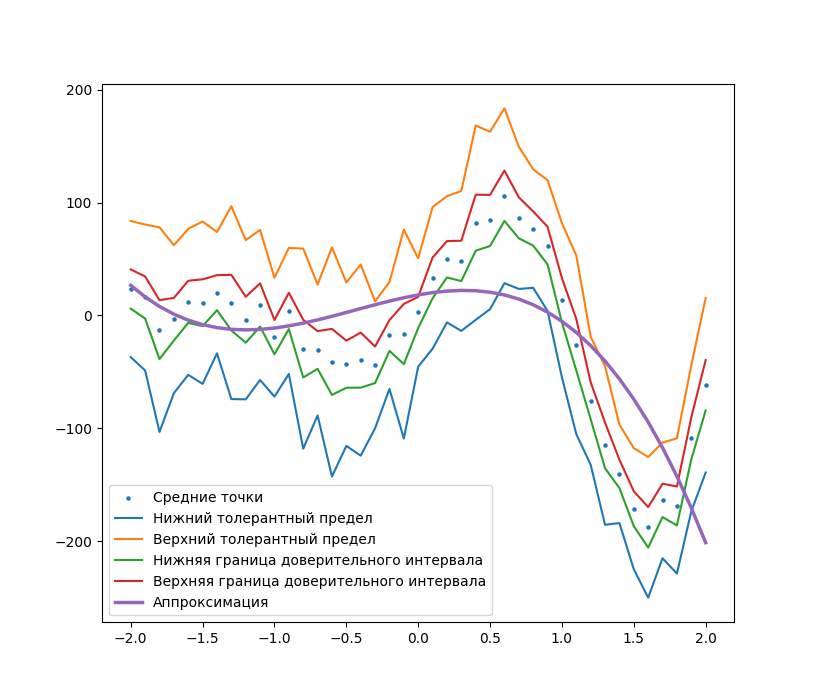
**Степень полинома *q* = 1**

  
Рис. 2 Степень *q* = 1

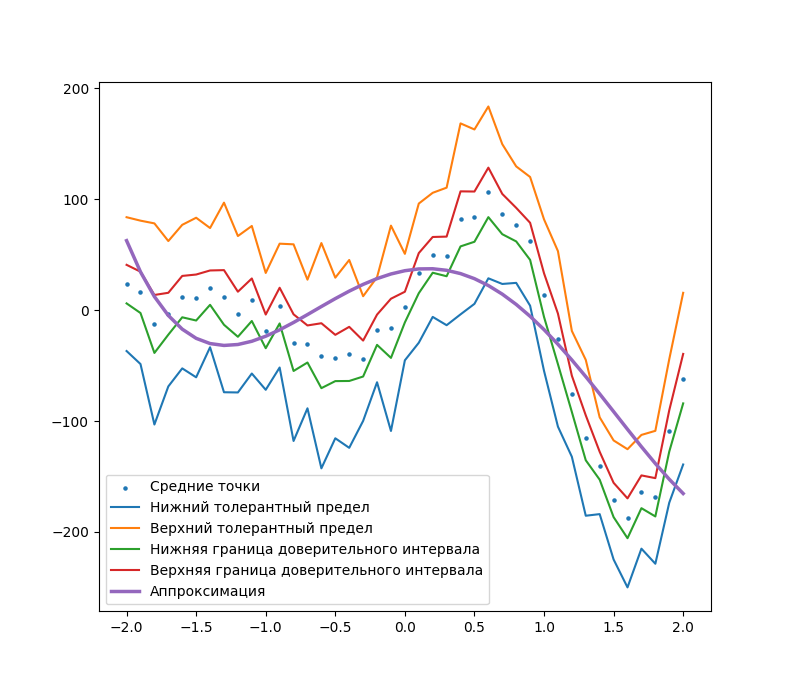
**Степень полинома *q* = 2**

  
Рис. 3 Степень *q* = 2

**Степень полинома *q* = 3**

  
Рис. 4 Степень *q* = 3

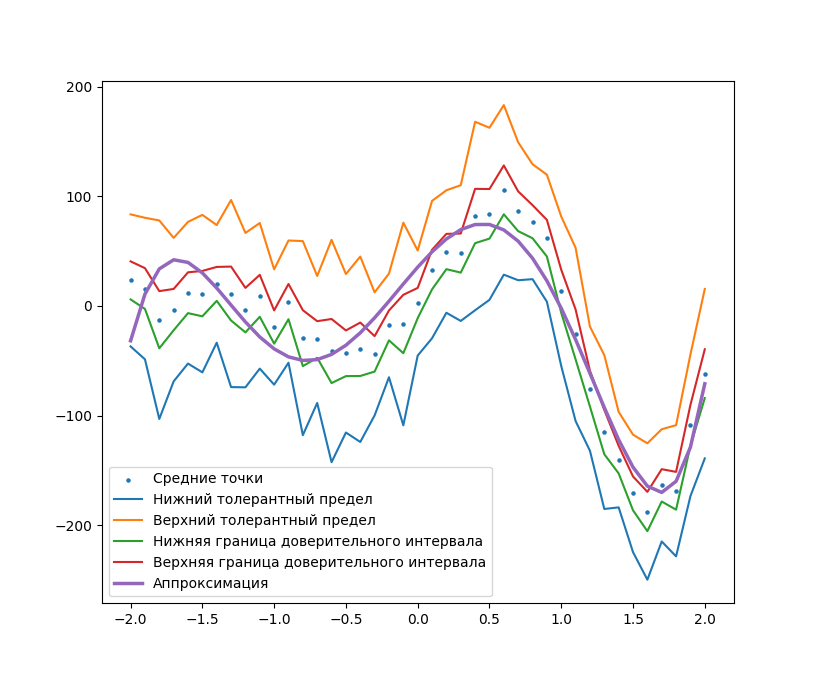
**Степень полинома *q* = 4**

  
Рис. 5 Степень *q* = 4

**Степень полинома *q* = 5**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| 6.19 | -4.30\*10-14 | -6.92 | 4.22\*10-14 | 1.49 | -8.46\*10-15 |
| -4.30\*10-14 | 24.2 | 8.14\*10-14 | -20 | -2.09\*10-14 | 3.94 |
| -6.92 | 8.14\*10-14 | 13.8 | -7.87\*10-14 | -3.55 | 1.56\*10-14 |
| 4.22\*10-14 | -21 | -7.86\*10-14 | 21.5 | 2\*10-14 | -4.4 |
| 1.49 | -2.09\*10-14 | -3.55 | 2\*10-14 | 9.89\*10-1 | -3.96\*10-15 |
| -8.46\*10-15 | 3.94 | 1.56\*10-14 | -4.4 | -3.96\*10-15 | 0.945 |

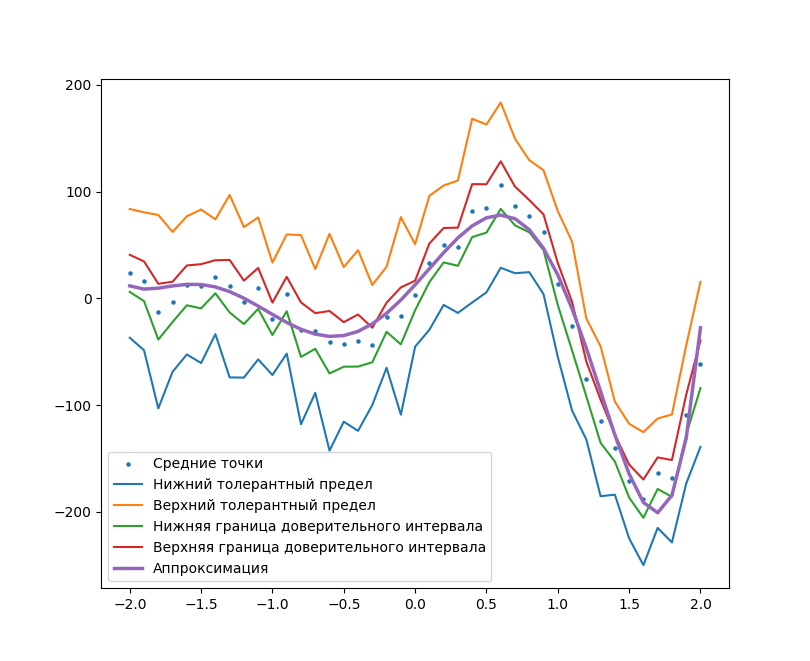
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| 1,000 | -0,003 | -0,971 | 0,002 | 0,953 | -0,001 |
| -0,003 | 1,000 | -0,008 | -0,997 | 0,010 | 0,993 |
| -0,971 | -0,008 | 1,000 | 0,005 | -0,998 | -0,003 |
| 0,002 | -0,997 | 0,005 | 1,000 | -0,006 | -0,999 |
| 0,953 | 0,010 | -0,998 | -0,006 | 1,000 | 0,004 |
| -0,001 | 0,993 | -0,003 | -0,999 | 0,004 | 1,000 |

  
Рис. 6 Степень *q* = 5

**Степень полинома *q* = 6**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sa | | | | | | |
| 8,46 | -1,01E-13 | -1,83E+01 | 8,90E-14 | 9,67 | -1,67E-14 | -1,44 |
| -1,01E-13 | 2,43E+01 | 3,73E-13 | -2,10E+01 | -2,30E-13 | 3,94E+00 | 3,68E-14 |
| -1,83E+01 | 3,73E-13 | 7,12E+01 | -3,14E-13 | -4,48E+01 | 5,73E-14 | 7,24E+00 |
| 8,90E-14 | -2,10E+01 | -3,14E-13 | 2,15E+01 | 1,89E-13 | -4,40E+00 | -2,97E-14 |
| 9,67E+00 | -2,30E-13 | -4,48E+01 | 1,89E-13 | 3,06E+01 | -3,39E-14 | -5,20E+00 |
| -1,67E-14 | 3,94E+00 | 5,73E-14 | -4,40E+00 | -3,39E-14 | 9,46E-01 | 5,26E-15 |
| -1,44E+00 | 3,68E-14 | 7,24E+00 | -2,97E-14 | -5,20E+00 | 5,26E-15 | 9,13E-01 |

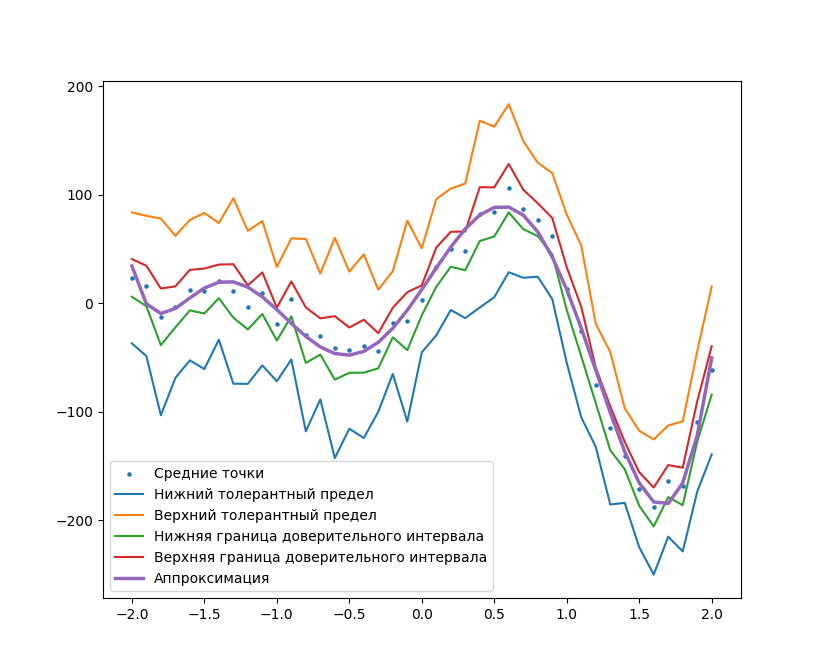
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
| 1,000 | 0,002 | -0,983 | -0,001 | 0,972 | 0,001 | -0,966 |
| 0,002 | 1,000 | -0,006 | -0,997 | 0,006 | 0,993 | -0,005 |
| -0,983 | -0,006 | 1,000 | 0,003 | -0,999 | -0,002 | 0,997 |
| -0,001 | -0,997 | 0,003 | 1,000 | -0,003 | -0,999 | 0,003 |
| 0,972 | 0,006 | -0,999 | -0,003 | 1,000 | 0,002 | -1,000 |
| 0,001 | 0,993 | -0,002 | -0,999 | 0,002 | 1,000 | -0,002 |
| -0,966 | -0,005 | 0,997 | 0,003 | -1,000 | -0,002 | 1,000 |

  
Рис. 7 Степень *q* = 6

**Степень полинома *q* = 7**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sa | | | | | | | |
| 8,46E+00 | -4,06E-13 | -1,83E+01 | 7,48E-13 | 9,67E+00 | -3,65E-13 | -1,44E+00 | 5,18E-14 |
| -4,06E-13 | 5,52E+01 | 1,31E-12 | -8,78E+01 | -7,62E-13 | 3,92E+01 | 1,17E-13 | -5,24E+00 |
| -1,83E+01 | 1,31E-12 | 7,12E+01 | -2,35E-12 | -4,48E+01 | 1,13E-12 | 7,24E+00 | -1,60E-13 |
| 7,48E-13 | -8,78E+01 | -2,35E-12 | 1,66E+02 | 1,34E-12 | -8,08E+01 | -2,03E-13 | 1,13E+01 |
| 9,67E+00 | -7,62E-13 | -4,48E+01 | 1,34E-12 | 3,06E+01 | -6,42E-13 | -5,20E+00 | 9,03E-14 |
| -3,65E-13 | 3,92E+01 | 1,13E-12 | -8,08E+01 | -6,42E-13 | 4,13E+01 | 9,67E-14 | -6,00E+00 |
| -1,44E+00 | 1,17E-13 | 7,24E+00 | -2,03E-13 | -5,20E+00 | 9,67E-14 | 9,13E-01 | -1,36E-14 |
| 5,18E-14 | -5,24E+00 | -1,60E-13 | 1,13E+01 | 9,03E-14 | -6,00E+00 | -1,36E-14 | 8,91E-01 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Ra | | | | | | | |
| 1,0000 | 0,0001 | -0,9830 | 0,0004 | 0,9722 | -0,0006 | -0,9656 | 0,0006 |
| 0,0001 | 1,0000 | -0,0003 | -0,9968 | 0,0003 | 0,9928 | -0,0003 | -0,9898 |
| -0,9830 | -0,0003 | 1,0000 | -0,0010 | -0,9985 | 0,0014 | 0,9965 | -0,0016 |
| 0,0004 | -0,9968 | -0,0010 | 1,0000 | 0,0009 | -0,9992 | -0,0009 | 0,9980 |
| 0,9722 | 0,0003 | -0,9985 | 0,0009 | 1,0000 | -0,0014 | -0,9996 | 0,0016 |
| -0,0006 | 0,9928 | 0,0014 | -0,9992 | -0,0014 | 1,0000 | 0,0013 | -0,9997 |
| -0,9656 | -0,0003 | 0,9965 | -0,0009 | -0,9996 | 0,0013 | 1,0000 | -0,0015 |
| 0,0006 | -0,9898 | -0,0016 | 0,9980 | 0,0016 | -0,9997 | -0,0015 | 1,0000 |

  
Рис. 8 Степень *q* = 7

Гипотеза о степени полинома прошла проверку, как только созданная функция уложилась в границы доверительных интервалов.

**Степень полинома *q = k – 1 = 40***

Характеристики коэффициентов получаются очень объёмными:

Изображение выглядит как текст, шаблон

Автоматически созданное описание  
Рис. 9 Матрицы, характеризующие полином степени *q = 40*

Полная таблица будет идти вместе с файлом.

Число обусловленности матрицы Sa также очень большое:

Изображение выглядит как диаграмма, График, линия, текст

Автоматически созданное описание  
Рис. 10 Степень полинома *q = 40*

Аппроксимация при большой степени полинома очевидно уступает по точности аппроксимации полиномом меньшей степени, что может быть объяснено слишком большой обусловленностью матрицы , для которой , тогда как для *q = 7* .

В итоге можно сделать вывод, что бездумное повышение степени полинома в погоне за увеличением точности аппроксимации является неэффективным и одним из наименее точных способов решить задачу.

# Аппроксимация другими способами

**Аппроксимация прямой**

Изображение выглядит как текст, снимок экрана, линия, диаграмма

Автоматически созданное описаниеРис. 11 Линейная аппроксимация разными способами

Полученные функции:

regress:

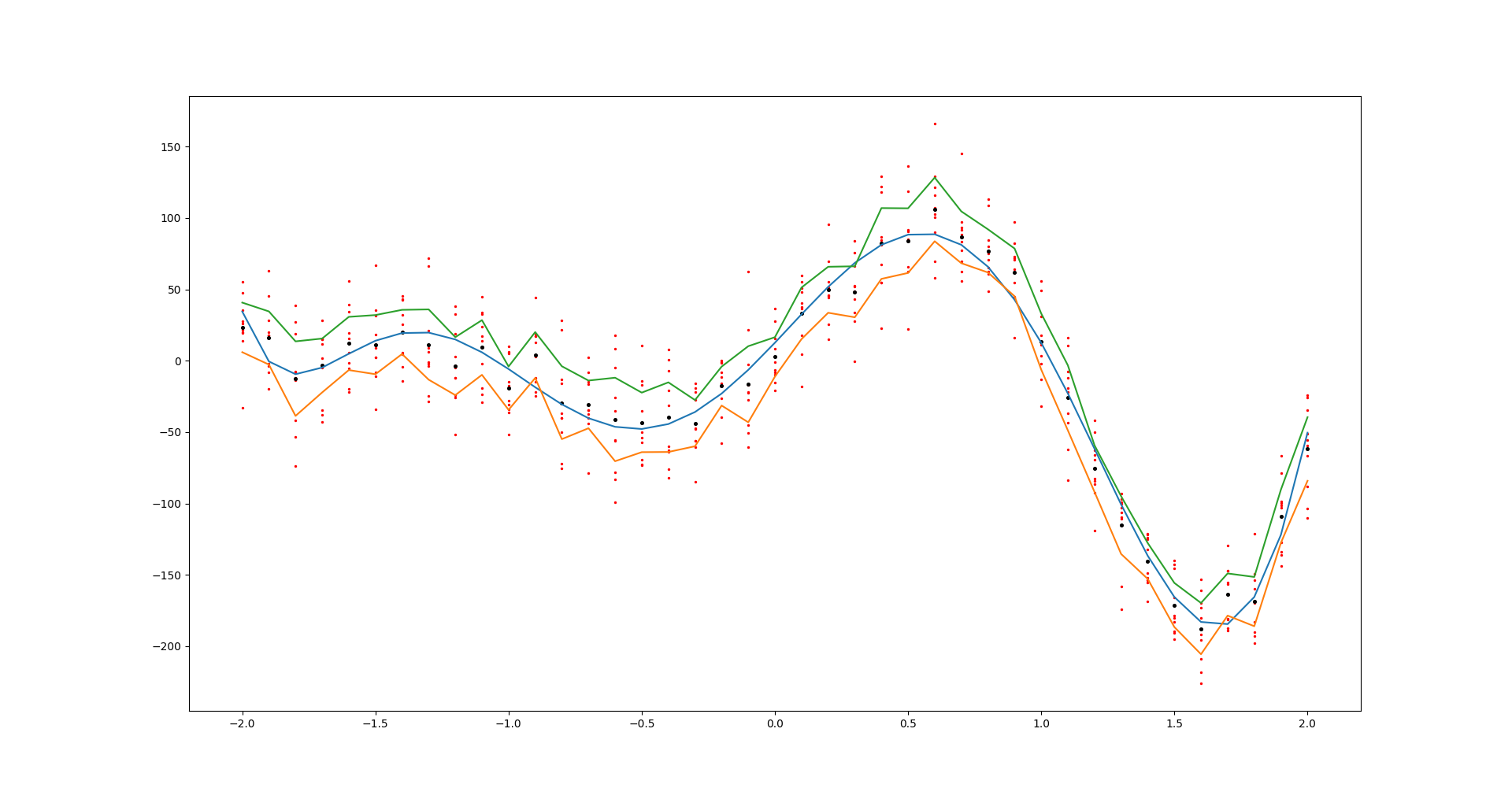
robust:

polyfit:

ridge:

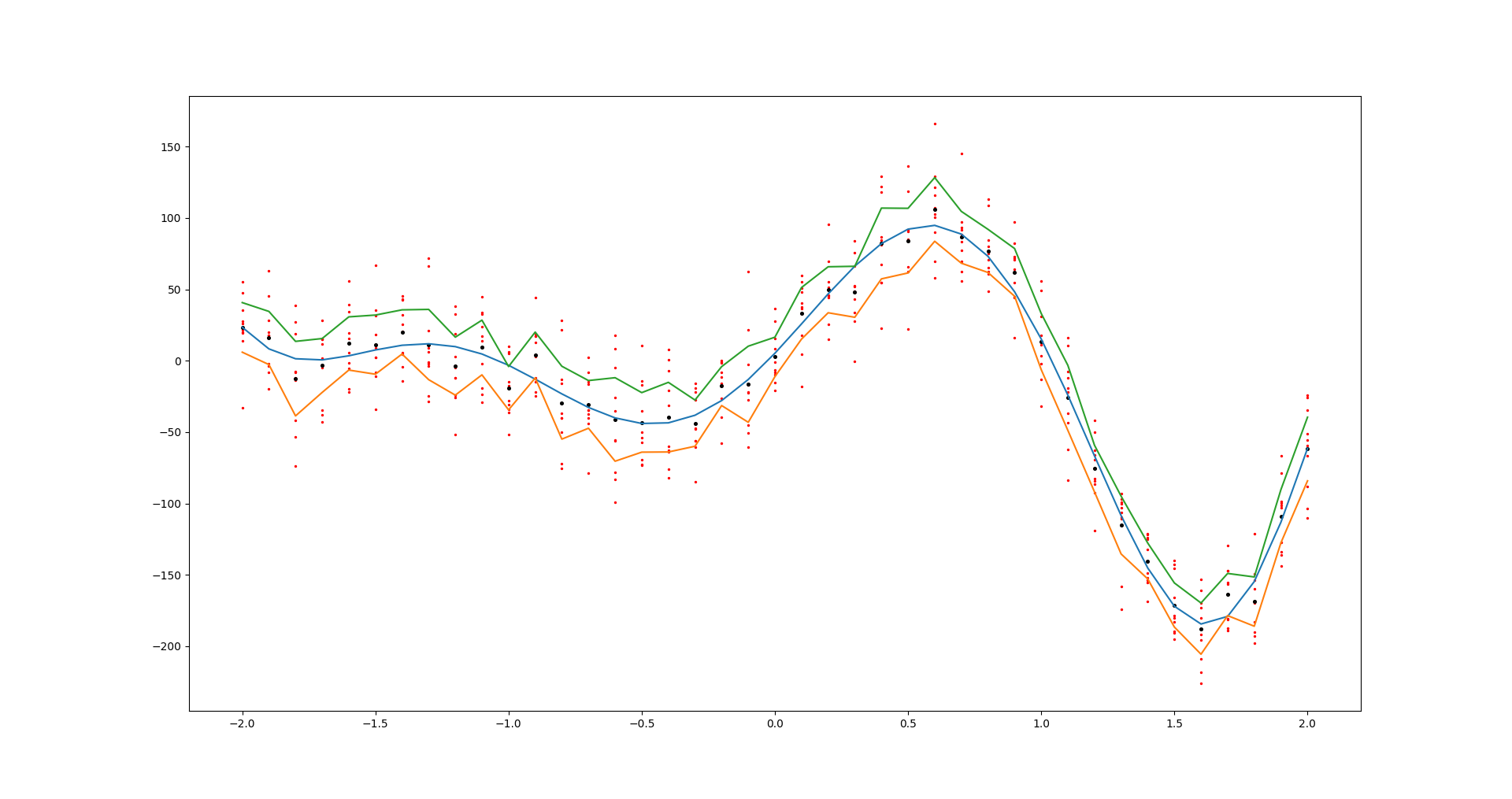
Из графика и уравнений видно, что методы regress и polyfit дали одинаковый результат, ridge довольно близок к ним, а robust немного отличается от остальных. Однако сделать вывод какой метод приближает функцию лучше не представляется возможным, потому что изначальная зависимость нелинейная.

**Полиномиальная аппроксимация**

  
Рис. 12 Аппроксимации с помощью polyfit при *q = 7*

Аппроксимация полиномом 7 степени является приемлемой по точности, так как большинство точек попадают в доверительный интервал.

Попробуем увеличить степень.

  
Рис. 13 Аппроксимации с помощью polyfit при *q = 8*

Полином 8 степени аппроксимирует функцию немного точнее.

При увеличении степени полинома вид аппроксимирующей кривой не изменяется примерно до 25 степени, а дальше становится все ближе и ближе к средним точкам, то есть дает практически тот результат, что получится при простейшей интерполяции. В связи с этим можно сделать вывод, аппроксимация полиномом 8-й степени является оптимальной.

**Кусочная полиномиальная аппроксимация**

Изображение выглядит как текст, диаграмма, линия, График

Автоматически созданное описание  
Рис. 14 Линейная интерполяция (interp)

Изображение выглядит как текст, диаграмма, карта, линия

Автоматически созданное описание  
Рис. 15 Интерполирование по Эрмиту

Изображение выглядит как текст, диаграмма, линия, График

Автоматически созданное описание  
Рис. 16 Сплайн-интерполяция

Среди методов интерполяции в данном случае предпочтительнее интерполяция по Эрмиту либо сплайнами, они обе довольно гладкие.

**Нелинейная аппроксимация**

Для нелинейной аппроксимации на Python будет использоваться scipy.optimize.curve\_fit(). Нелинейная функция:

Степень берётся как и в полиномиальной аппроксимации *q = 7*.

– начальный вектор коэффициентов используемый для аппроксимации

Изображение выглядит как текст, диаграмма, снимок экрана, График

Автоматически созданное описание  
Рис. 17 Нелинейная аппроксимация

Нелинейная аппроксимация ведёт себя схожим образом с полиномиальной.

# Вывод

В ходе выполнения работы была проведена аппроксимация МНК с проверкой гипотезы о степени полинома, были исследованы ее параметры. Первым прошедшим гипотезу по критерию Фишера оказался полином степени q = 7, он был сравнен с полиномом степени q = k-1 = 40. Последний же полином оказался катастрофически неточным. Подобное поведения объясняется плохой обусловленностью матрицы , обращаемой в процессе получения коэффициентов полинома. Были произведены аппроксимации прямыми линиями, результаты которых являются малопригодными вследствие явно нелинейной природы исходных данных. Результаты кусочно-полиномиальных интерполяций хотя и оказались намного лучше линейных, но все равно были недостаточно точны, так как исходные данные были получены многократными измерениями с погрешностью. Принимая во внимание такой фактор как трудоемкость вычислений, можно сказать, что оптимальным будет вариант с интерполяцией сплайнами или интерполяцией полиномами Эрмита, так как эти методы являются малозатратными и в то же время позволяют повторить вид исходной зависимости. Исходя из желания получить наиболее точное решение, можно сделать вывод что аппроксимации многократных экспериментальных измерений с погрешностью лучше всего получать с помощью МНК, ОМНК или методов нелинейной регрессии, использующими в качестве нелинейной функции зависимость, специально подобранную для конкретной ситуации. Какой метод выбирать для решения конкретной задачи – вопрос, не имеющий однозначного ответа, так как в подобных ситуациях всегда необходимо находить баланс между точностью и трудоемкостью вычисления решения.

# Приложение

import re  
  
import matplotlib  
import numpy as np  
import matplotlib.pyplot as plt  
import scipy  
import openpyxl  
from scipy.optimize import curve\_fit  
from sklearn.linear\_model import Ridge  
  
matplotlib.use('TkAgg')  
wb = openpyxl.Workbook()  
sheet = wb.active  
  
data\_text = open('Task\_2b.txt', 'r')  
  
nx = int(re.findall(r'\d+', data\_text.readline().strip())[0])  
ny = int(re.findall(r'\d+', data\_text.readline().strip())[0])  
  
\_ = data\_text.readline().strip()  
  
Y = []  
x\_points = []  
  
for i in range(nx):  
 line = list(map(float, re.findall(r'[-+]?(?:\d\*\.\*\d+)', data\_text.readline().strip())))  
 x\_points.append(line[0])  
 Y.append(line[1:])  
  
data\_text.close()  
  
k = 2.535 # толерантный множитель  
t = 2.31 # коэффициент стьюдента  
  
i = 0  
sall = []  
avg\_y = []  
lcis = []  
ucis = []  
ltols = []  
utols = []  
  
for i in range(nx):  
 avg\_yi = np.mean(Y[i])  
 avg\_y.append(avg\_yi)  
 variance = 1 / (ny - 1) \* sum([(yk - avg\_yi) \*\* 2 for yk in Y[i]])  
 sall.append(variance)  
 lci = avg\_yi - t \* (variance \*\* (1 / 2)) / (ny \*\* (1 / 2))  
 lcis.append(lci)  
 uci = avg\_yi + t \* (variance \*\* (1 / 2)) / (ny \*\* (1 / 2))  
 ucis.append(uci)  
 ltol = avg\_yi - (variance \*\* (1 / 2)) \* k  
 ltols.append(ltol)  
 utol = avg\_yi + (variance \*\* (1 / 2)) \* k  
 utols.append(utol)  
  
 i += 1  
  
kochren = max(sall) / sum(sall)  
  
se2 = 1 / nx \* sum(sall) # средняя оценка дисперсии  
q = 0 # начальная степень полинома  
Fp = 1000 # начальное значение Fp критерия  
while Fp > scipy.stats.f.ppf(0.95, nx - q - 1, ny - 1):  
 X = np.vander(x\_points, N=q + 1, increasing=True) # матрица X  
 a = np.linalg.inv(X.T @ X) @ X.T @ np.array(avg\_y) # оценка коэффициентов  
 for i in range(len(a)):  
 sheet.cell(row=1, column=i + 1, value=a[i])  
 Sa = 1 / ny \* se2 \* (np.linalg.inv(X.T @ X)) # ковариационная матрица оценок коэффициентов  
 print(np.linalg.cond(Sa))  
 for i in range(len(Sa)):  
 for j in range(len(Sa[i])):  
 sheet.cell(row=i + 3, column=j + 1, value=Sa[i][j])  
 Ra = np.corrcoef(Sa, rowvar=False)  
 Fp = 1 / (nx - q - 1) \* ny \* se2 \*\* (-1) \* (X @ a - np.array(avg\_y)).T @ (X @ a - np.array(avg\_y))  
 for i in range(len(Ra)):  
 for j in range(len(Ra[i])):  
 sheet.cell(row=i + len(Sa), column=j + 1, value=Ra[i][j])  
 wb.save('sa.xlsx')  
  
 approximation = X @ a  
  
 plt.scatter(x\_points, avg\_y, label='Средние точки', s=5)  
 plt.plot(x\_points, ltols, label='Нижний толерантный предел')  
 plt.plot(x\_points, utols, label='Верхний толерантный предел')  
 plt.plot(x\_points, lcis, label='Нижняя граница доверительного интервала')  
 plt.plot(x\_points, ucis, label='Верхняя граница доверительного интервала')  
 plt.plot(x\_points, approximation, label='Аппроксимация', lw=2.5)  
 plt.legend()  
 plt.show()  
  
 q += 1  
  
f = False  
for i in range(ny):  
 y = []  
 for j in range(nx):  
 y.append(Y[j][i])  
 if not f:  
 plt.scatter(x\_points, y, color='red', s=2, label='Измерения')  
 f = True  
 else:  
 plt.scatter(x\_points, y, color='red', s=2)  
  
plt.scatter(x\_points, avg\_y, label='Точки аппроксимации', s=7, color='black')  
  
regress = scipy.stats.linregress(x\_points, avg\_y)  
print(f'regress: y = {regress.slope} \* x + ({regress.intercept}) ')  
approximation = []  
for x in x\_points:  
 approximation.append(regress.slope \* x + regress.intercept)  
plt.plot(x\_points, approximation, label='regress', color='blue')  
  
robust = scipy.stats.siegelslopes(avg\_y, x\_points)  
print(f'robust: y = {robust.slope} \* x + ({robust.intercept}) ')  
approximation = []  
for x in x\_points:  
 approximation.append(robust.slope \* x + robust.intercept)  
plt.plot(x\_points, approximation, label='robust', color='orange')  
  
polyfit = np.polynomial.polynomial.Polynomial.fit(x\_points, avg\_y, 1)  
print(polyfit)  
approximation = []  
for x in x\_points:  
 approximation.append(polyfit(x))  
plt.plot(x\_points, approximation, label='polyfit', linestyle='dashed', dashes=(5, 5), color='red')  
  
model\_ridge = Ridge(alpha=1.0)  
model\_ridge.fit(np.array(x\_points).reshape(-1, 1), avg\_y)  
approximation = model\_ridge.predict(np.array(x\_points).reshape(-1, 1))  
print(f'ridge: y = {model\_ridge.coef\_[0]} \* x + ({model\_ridge.intercept\_})')  
plt.plot(x\_points, approximation, label='ridge', color='green')  
  
plt.legend()  
plt.show()  
  
polyfit = np.polynomial.polynomial.Polynomial.fit(x\_points, avg\_y, 40)  
print(polyfit)  
approximation = []  
for x in x\_points:  
 approximation.append(polyfit(x))  
plt.plot(x\_points, approximation, label='polyfit')  
plt.plot(x\_points, lcis, label='Нижняя граница доверительного интервала')  
plt.plot(x\_points, ucis, label='Верхняя граница доверительного интервала')  
plt.show()  
  
x\_smooth = np.linspace(-2, 2, 200)  
approximation = np.interp(x\_smooth, x\_points, avg\_y)  
plt.plot(x\_smooth, approximation)  
plt.plot(x\_points, lcis, label='Нижняя граница доверительного интервала')  
plt.plot(x\_points, ucis, label='Верхняя граница доверительного интервала')  
plt.show()  
  
approximation = scipy.interpolate.PchipInterpolator(np.array(x\_points), np.array(avg\_y))  
x\_smooth = np.linspace(-2, 2, 200)  
plt.plot(x\_smooth, approximation(x\_smooth))  
plt.plot(x\_points, lcis, label='Нижняя граница доверительного интервала')  
plt.plot(x\_points, ucis, label='Верхняя граница доверительного интервала')  
plt.show()  
  
approximation = scipy.interpolate.CubicSpline(x\_points, avg\_y)  
x\_smooth = np.linspace(-2, 2, 200)  
plt.plot(x\_smooth, approximation(x\_smooth))  
plt.plot(x\_points, lcis, label='Нижняя граница доверительного интервала')  
plt.plot(x\_points, ucis, label='Верхняя граница доверительного интервала')  
plt.show()  
  
  
def model\_func(x, alpha, beta, a7, a6, a5, a4, a3, a2, a1, a0):  
 return (np.sin(alpha \* x) + beta) \* (a7 \* x\*\*7 + a6 \* x\*\*6 + a5 \* x\*\*5 + a4 \* x\*\*4 + a3 \* x\*\*3 + a2 \* x\*\*2 + a1 \* x + a0)  
  
  
initial\_guess = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]  
params = curve\_fit(model\_func, x\_points, avg\_y, p0=initial\_guess)[0]  
print(params)  
alpha, beta, a7, a6, a5, a4, a3, a2, a1, a0 = params  
x\_smooth = np.linspace(-2, 2, 200)  
y = model\_func(x\_smooth, alpha, beta, a7, a6, a5, a4, a3, a2, a1, a0)  
plt.plot(x\_smooth, y)  
plt.plot(x\_points, lcis, label='Нижняя граница доверительного интервала')  
plt.plot(x\_points, ucis, label='Верхняя граница доверительного интервала')  
plt.show()