Математические эксперименты в среде SMath Studio

Очков В.Ф.

Один из афоризмов Козьмы Пруткова гласит: «*Бросая в воду камешки, смотри на круги, ими образуемые; иначе такое бросание будет пустою забавою*». А давайте бросать не камешки, а точки и не в воду, а на плоскость и смотреть на… некоторые математические задачи «ими образуемые».

На рисунке 1 (первая строка) показано, как в среде свободно распространяемой математической программы SMath Studio (www.smath.com, [1]) функция `Random` возвращает два вектора, содержащие *n* случайных чисел в интервале от 0 до 1, а функция `augment` соединяет эти векторы в матрицу с двумя столбцами и `n` строками, отображаемую на графике в квадрате со стороной, равной единице. По умолчанию на графике будет нарисована некая каляка-маляка – отрезки прямых, соединяющих точки в порядке их номеров (рис. 1a). Маленькие кружочки без отрезков прямых линий (рис. 1b) появятся после несложного форматирования графика. Ломаную линию, состоящую из отрезков прямых (рис. 1a) можно отформатировать так, чтобы она превратилась в отрезки кубической параболы, плавно переходящих друг в друга (рис. 1c). Таким форматированием мы воспользуемся, когда будем формировать условия новой интересной задачи.

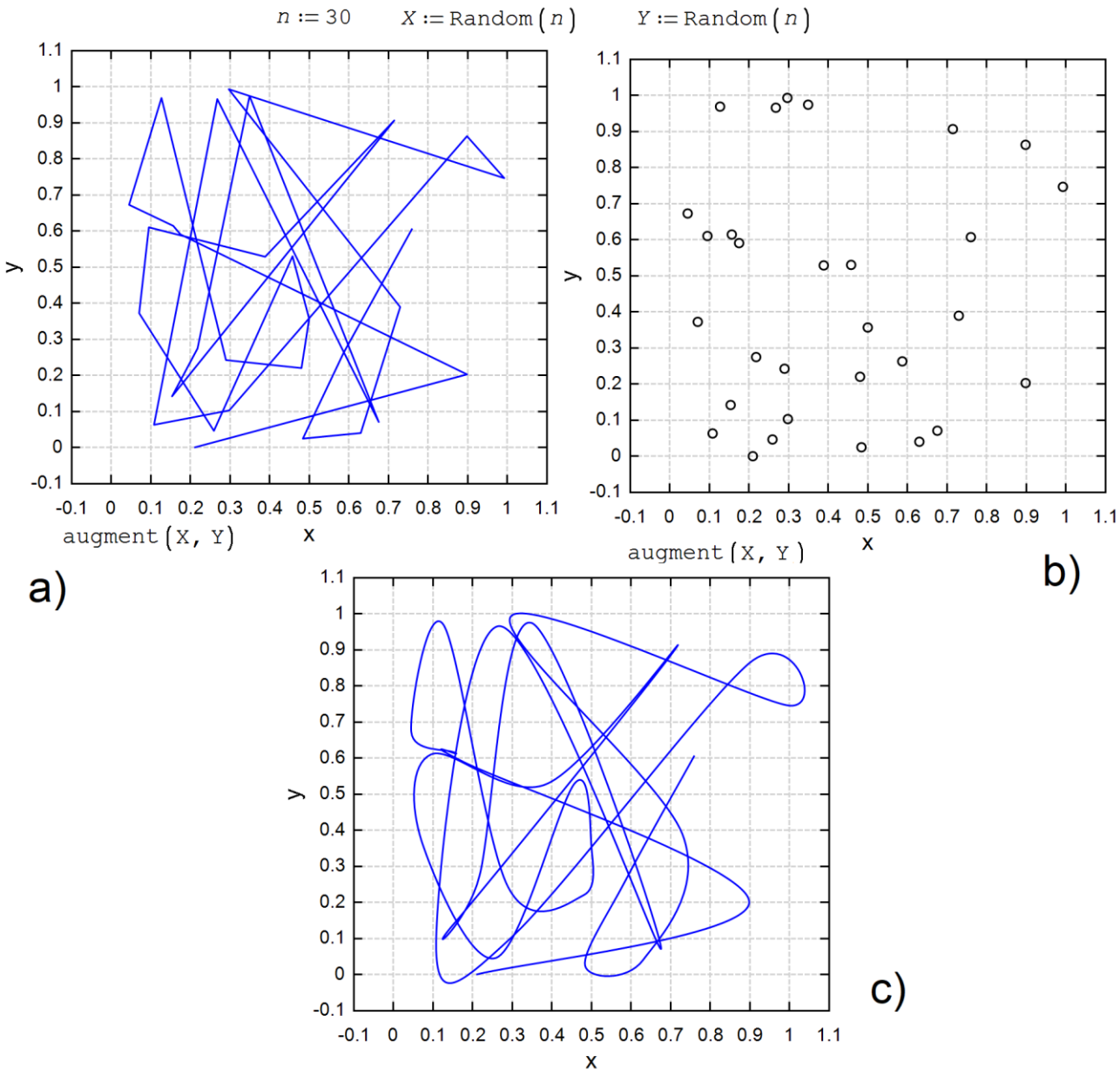


Рис. 1. Случайные точки на плоскости

Задача 1. Гибридное решение задачи коммивояжёра

Точки на рисунке 1b необходимо соединить отрезками прямых линий так, чтобы они отмечали *маршрут коммивояжёра* – человека, которому поручено обойти все точки, побывав в каждой по одному разу с минимизацией пройденного пути. Таких людей мы ежечасно видим в наших городах на электровелосипедах с жёлтыми квадратными сумками на плечах. Это наши кормилицы!

Разработано множество алгоритмов решения этой задачи – *задачи коммивояжёра*. Если расставить их в порядке возрастания времени выполнения на компьютере, то на левом фланге окажется простейший *метод ближайшего соседа*, а на правом – *метод прямого перебора*. Первым метод подразумевает, что коммивояжёр из очередной точки переходит в ближайшую, а второй сводится к генерации всех возможных маршрутов обхода точек и выбор из них наикратчайшего. Метод ближайшего соседа при числе точек даже меньше десяти дает весьма приближенный результат, требующий дальнейшей оптимизации. Прямой перебор в принципе должен дать точный ответ, но… В статье Википедии, посвященной этой задаче [2], первой картинкой сразу показано, что при 14 точках число вариантов их обхода превысит… 40 миллиардов.

А давайте совместим эти два крайних метода и получим некий *гибридный* метод, дающий приемлемый результат даже при нескольких десятках точек.

Для реализации метода ближайшего соседа нам потребуются создать на языке программирования SMath две вспомогательные функции. Первая с именем M (рис. 2) возвращает квадратную матрицу расстояний между точками на плоскости[1]. Главная диагональ этой матрицы содержит бесконечное значение. Это будет означать, что пути из точки в эту же точку не существует. Бесконечное значение будет также присваиваться соответствующему участку прямолинейного пути после того, как он будет пройден.

[1] Автор помнит подмосковные автобусы, где висела такая матрица – таблица, по которой можно было узнать, сколько с вас возьмет кондуктор за проезд из пункта А в пункт Б.

$$M(X, Y) := \left| \begin{array}{l} \text{for } i \in [1..\text{length}(X)] \\ \quad \text{for } j \in [1..\text{length}(X)] \\ \quad\quad M_{i,j} := \text{if } i = j \\ \quad\quad\quad\quad \infty \\ \quad\quad\quad\quad \text{else} \\ \quad\quad\quad\quad\quad \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2} \\ M \end{array} \right.$$

$$M\left(\begin{bmatrix}1\\2\\7\\2\\1\end{bmatrix}, \begin{bmatrix}4\\5\\0\\5\\3\end{bmatrix}\right) = \begin{bmatrix} \infty & 1.414 & 7.211 & 1.414 & 1 \\ 1.414 & \infty & 7.071 & 0 & 2.236 \\ 7.211 & 7.071 & \infty & 7.071 & 6.708 \\ 1.414 & 0 & 7.071 & \infty & 2.236 \\ 1 & 2.236 & 6.708 & 2.236 & \infty \end{bmatrix}$$

Рис. 2. Создание матрицы расстояний между точками на плоскости

Вторая вспомогательная пользовательская функция с именем i_{min} (рис. 3) возвращает номер элемента матрицы с минимальным значением.

$$i_{min}(V) := \left| \begin{array}{l} \text{for } i \in [1..\text{length}(V)] \\ \quad \text{if } V_i = \min(V) \\ \quad\quad i_{min} := i \\ i_{min} \end{array} \right. \qquad i_{min}\left(\begin{bmatrix}5\\6\\2\\1\\8\end{bmatrix}\right) = 4$$

Рис. 3. Поиск номера элемента вектора с минимальным значением

На рисунках 2 и 3 показаны сами функции и примеры их вызова для тестирования.

Функция с именем *Way* (рис. 4) решает поставленную задачу – возвращает вектор, содержащий номера точек в том порядке, какой требует метод ближайшего соседа с возвратом

в исходную точку. Номер стартовой точки i_{start} задается первым аргументом функции Way. Координаты точек – это аргументы X и Y.

```
Way(i_start, X, Y) :=
    M := M(X, Y)
    Way_1 := i_start
    for i ∈ [2..n]
        for j ∈ [1..n]
            S_j := M_{Way_{i-1}, j}
        Way_i := i_min(S)
        for j ∈ [1..i]
            M_{Way_i, Way_j} := ∞
    Way_{n+1} := Way_1
    Way
```

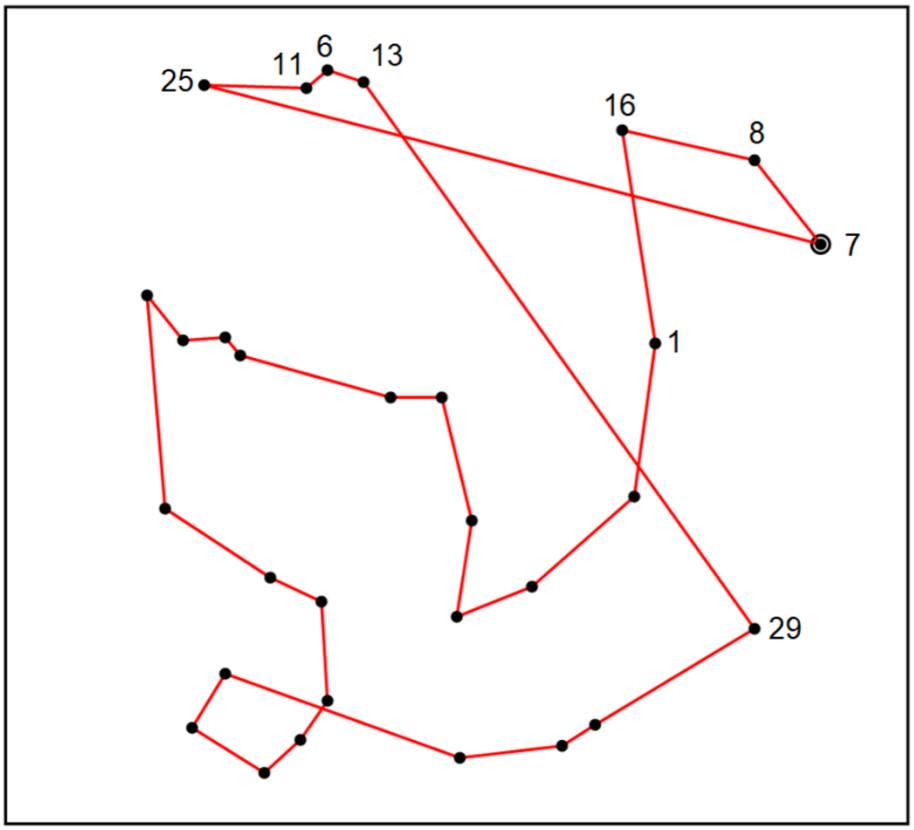
Рис. 4. Решение задачи коммивояжёра методом ближайшего соседа

В середине программы замеряются расстояния от очередной точки до всех остальных точек – выбираются нужные элементы из матрицы M, которые помещаются в вектор S. Индекс его минимального элемента будет номером очередной точкой пути коммивояжёра: $Way_i := i_{min}(S)$. Далее «сжигаются мосты»: соответствующим элементам матрицы расстояний M присваиваются значение бесконечности. Это, как отмечено ранее, исключает повторное использование данного отрезка пути из точки i в точку j.

Если за стартовую точку взять седьмую, то путь коммивояжёра будет с петлями, удлиняющими маршрут – см. рис. 5. И это понятно – метод ближайшего соседа относится к *жадной* группе методов решения задач: локальная оптимизация приводит к проблемам при итоговой оптимизации.

$i_{start} := 7 \quad Путь := Way\left(i_{start}, X, Y\right)$

$Путь^{T} = [\,7\ 8\ 16\ 1 \cdots 29\ 13\ 6\ 11\ 25\ 7\,]$



$$\begin{cases} augment\left(X, Y, "."\right) \\ augment\left(X_{Путь_{[1..n+1]}}, Y_{Путь_{[1..n+1]}}\right) \\ \left[X_{i_{start}} \; Y_{i_{start}} \; "o" \; 10 \; "black"\right] \end{cases}$$

Рис. 5. Один из путей коммивояжёра

Примечание. Следует обратить внимание на вторую запись в аргументе графика на рис. 5. Там индексы у векторов X и Y – это не целочисленные скаляры, а новый вектор $Путь$, у которого индекс – это ещё один вектор с номерами точек. Вот какая получилась сложная конструкция, в которой часто путаются школьники и студенты. А есть в аргументе графика и в самом расчете текстовый индекс – см. переменную i_{start}, что ещё больше сбивает с толку школьников и студентов. Ввод конструкций, показанных на рисунках 3, 4 и 5, это хорошее упражнение по работе с векторами и матрицами!

Выбор другой стартовой точки может оказаться оптимальнее и из-за отсутствия петель. А давайте переберем все точки в роли стартовой и найдем точку, дающую самый короткий путь! Получится некий *гибридный метод решения задачи коммивояжёра*.

На рисунке 6 показана программа перебора всех точек и выбора оптимальной стартовой точки – точки под номером 17 (рис. 7a, где эта точка отмечена кружочком).

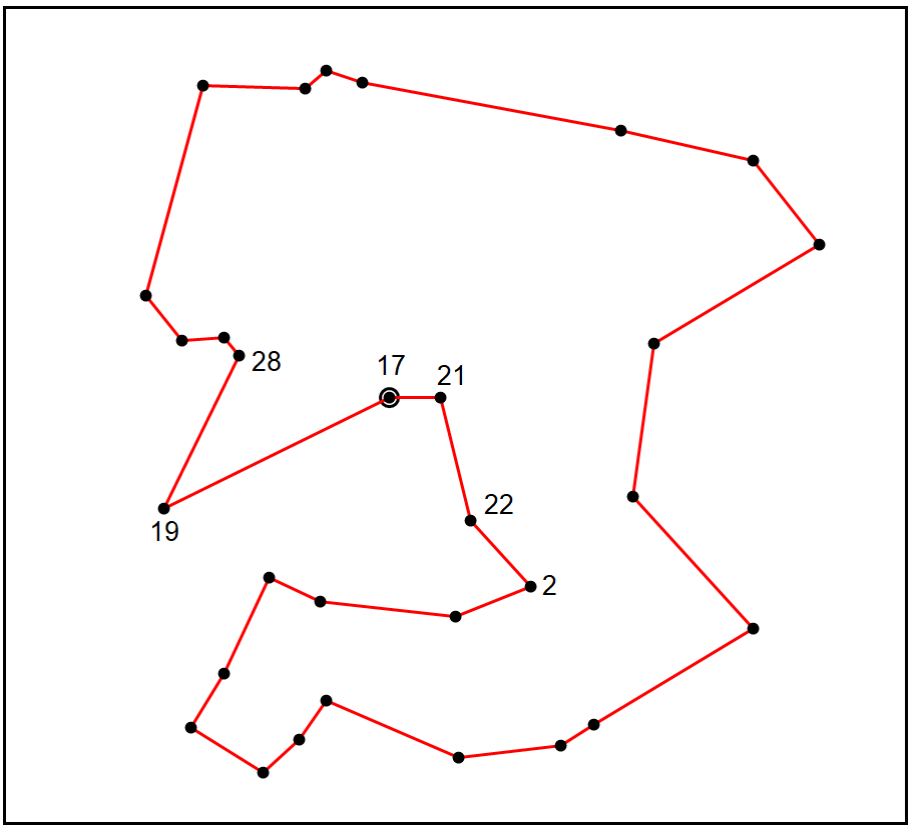
$$i_{start} := \left| \begin{array}{l} \Sigma S_{min} := \infty \\ \text{for } i_{start} \in [1..n] \\ \quad \left| \begin{array}{l} Путь := Way\left(i_{start}, X, Y\right) \\ \Sigma S := \sum_{i=1}^{n} \sqrt{\left(X_{Путь_i} - X_{Путь_{i+1}}\right)^2 + \left(Y_{Путь_i} - Y_{Путь_{i+1}}\right)^2} \\ \text{if } \Sigma S < \Sigma S_{min} \\ \quad \left| \begin{array}{l} i_{min} := i_{start} \\ \Sigma S_{min} := \Sigma S \end{array} \right. \end{array} \right. \\ i_{min} \end{array} \right. = 17$$

Рис. 6. Оптимизация пути коммивояжёра перебором

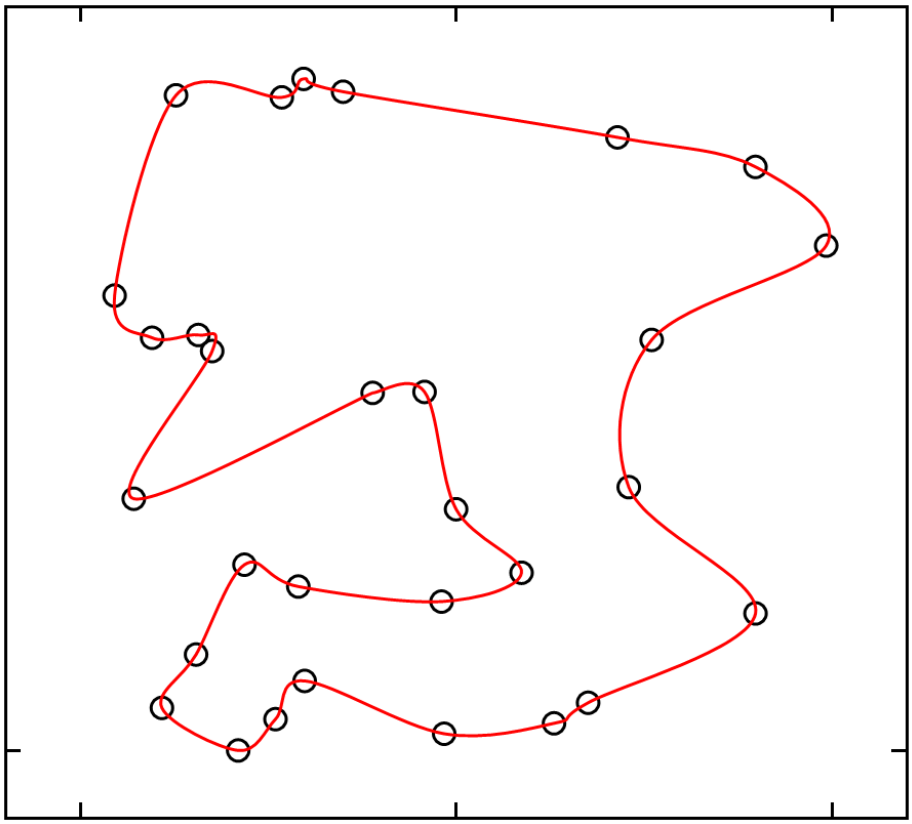
А что показано на рисунке 7b? Это путь коммивояжёра, но по отрезкам не *прямых*, а *кубических парабол*! Природа не любит острых углов! Представим себе, что наш развозчик товаров на электровелосипеде объезжает заказчиков так, чтобы не спешиваться и не разворачивать велосипед вручную чуть ли не на 180 градусов… Он едет по замысловатой гладкой кривой и на ходу, чтобы не терять времени сбрасывает упакованные товары у калитки дома заказчика. Путь может оказаться длиннее, но время в пути сократится. А время – это, как известно, деньги. Исходя из этого, можно сформировать новую, доныне неисследованную суперзадачу коммивояжёра – обойти все точки, но не по прямой, а по гладкой кривой, состоящей из отрезков кубической параболы, затратив при этом минимум времени при постоянной скорости. Такая гладкая кривая, кстати, хорошо известна в математике. Она используется для интерполяции сплайнами.

$$Путь := Way\left(i_{start}, X, Y\right)$$

$$Путь^{T} = \left[\begin{array}{cccccccccc} 17 & 21 & 22 & 2 & 23 & \cdots & 28 & 19 & 17 \end{array}\right]$$



a)



b)

Рис. 7. Путь коммивояжёра после оптимизации перебором: a) угловатый, b) гладкий

Координаты точек на плоскости задавались случайным образом – см. первую строку на рис. 1. Если нажимать клавишу F9 (пересчет – бросание новых камешков в воду), то может оказаться так, что и в оптимальном маршруте, найденном методом ближайшего соседа с перебором, будут петли – см левый график на рисунке 8. Можно, конечно, усложнить программу поиска решения, а можно не заморачиваться и в ручном режиме переставить некоторые элементы в векторе *Путь* – см. правый график на рис. 8.

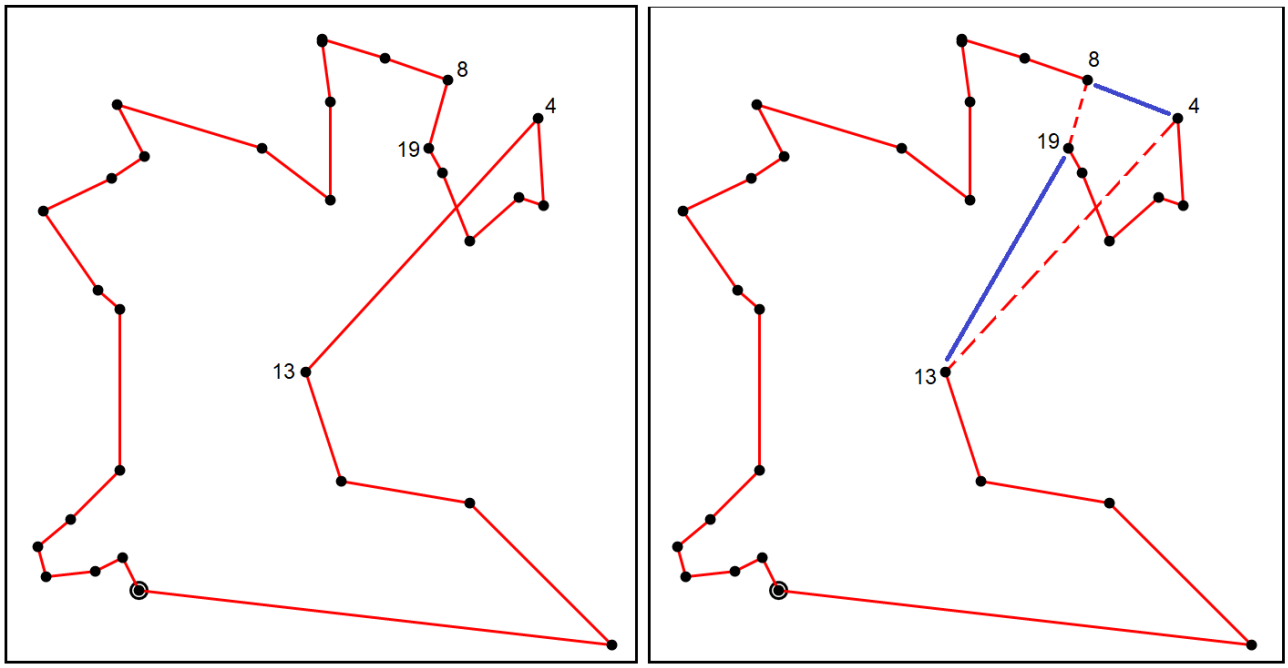


Рис. 8. Ручная корректировка пути коммивояжёра

Школьники или студенты, «наигравшись» описанной выше задачей, могут перейти к реализации в среде SMath более сложных алгоритмов решения задачи коммивояжёра, перечисленных в [2].

Интерлюдия. Число π и чёрный круг Малевича

Про чёрный квадрат Малевича знают все – даже те, кто совсем не разбирается в живописи. А вот про чёрный круг Малевича знают немногие. Автор узнал о нём только тогда, когда писал эту статью и сделал в интернете соответствующий поиск. А давайте нарисуем этот круг и заодно рассчитаем приближённое значение числа π!

На рисунке 9 показано, как в круг с единичным диаметром и с центром в начале координат бросают случайные камешки, пардон, чёрные точки. Есть такое направление изобразительного искусства под названием пуантилизм, основанное на технике письма мелкими точечными мазками почти чистых красок.

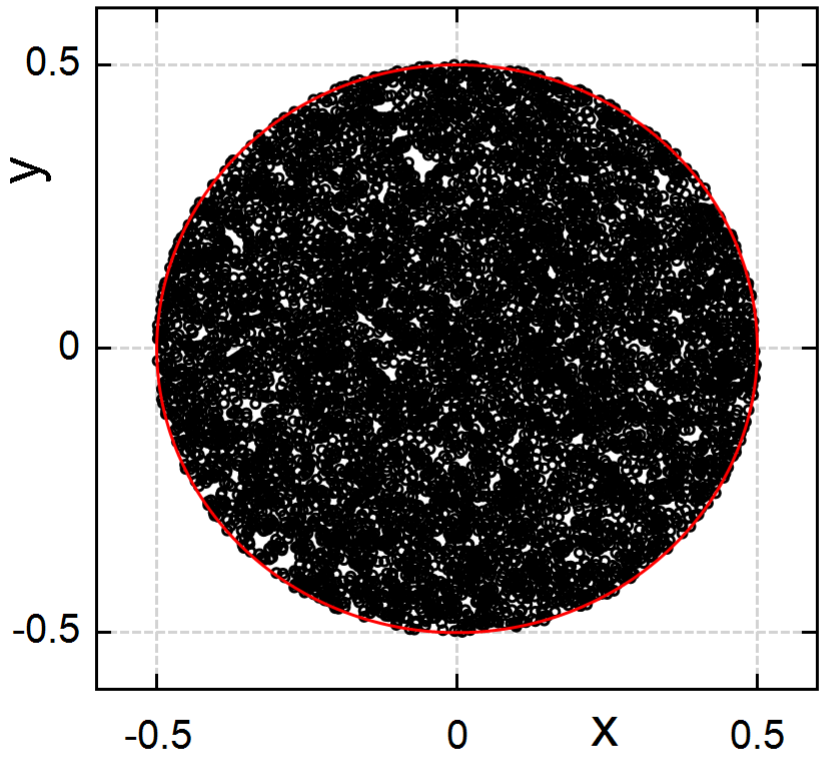
В программе с перебором (опять перебор!) из 10000 точек, бросаемых в квадрат с единичной стороной, отбираются те, какие попадают в круг с единичным диаметром. Отношение числа точек, попавших в круг, к общему числу точек хранит искомое приближенное значение числа π.

$n := 10000$ $X := \text{Random}(n) - 0.5$

$Y := \text{Random}(n) - 0.5$

$V :=$ | $no := 0$
for $i \in [1..n]$
if $X_i^2 + Y_i^2 < 0.5^2$
$[no := no + 1 \quad Xo_{no} := X_i \quad Yo_{no} := Y_i]$
$[Xo \; Yo \; no]$

$Xo := V_1$ $Yo := V_2$ $no := (V_3) = 7856$



$\begin{cases} \text{augment}(Xo, Yo, "o", 3) \\ x^2 + y^2 - 0.5^2 \end{cases}$

$4 \cdot \frac{no}{n} = 3.1424$ $\pi = 3.1416$

Рис. 9. Черный круг Малевича

Расчёт на рис. 9 – это по своей сути не определение числа π, а проверка качества генератора случайных чисел – функции `Random`.

Примечание. Настоящий чёрный квадрат Малевича отличают от подделок по трещинам лака прямоугольной формы. Кракелюром они называются. Наш чёрный круг тоже имеет некие трещины, но они, как и положено имеют круглую форму.

Задача 2. Эллипс, гипербола и примкнувшая к ним парабола

А может коммивояжёр объехать своих клиентов не по кубической параболе, а по кругу или по эллипсу?

Давайте решим такую задачу – отметим на плоскости пять точек и проведем через них гладкий путь коммивояжёра – кривую второго порядка: см. рис. 10. Для этого необходимо и достаточно создать квадратную матрицу размером шесть на шесть, заполнить её соответствующим образом и найти определитель, формирующий неявную функцию вида `f(x, y)`.

$X := \text{Random}(5) - 0.5 \quad Y := \text{Random}(5) - 0.5$

Матрицы

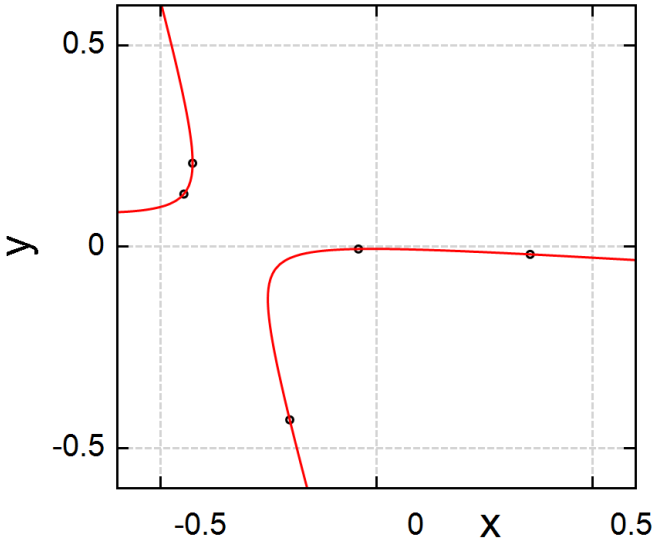
Определитель матрицы

$$f(x, y) := \begin{vmatrix} x^2 & x \cdot y & y^2 & x & y & 1 \\ X_1^2 & X_1 \cdot Y_1 & Y_1^2 & X_1 & Y_1 & 1 \\ X_2^2 & X_2 \cdot Y_2 & Y_2^2 & X_2 & Y_2 & 1 \\ X_3^2 & X_3 \cdot Y_3 & Y_3^2 & X_3 & Y_3 & 1 \\ X_4^2 & X_4 \cdot Y_4 & Y_4^2 & X_4 & Y_4 & 1 \\ X_5^2 & X_5 \cdot Y_5 & Y_5^2 & X_5 & Y_5 & 1 \end{vmatrix}$$

Вставка матрицы

Строки: 6

Столбцы: 6



augment(X, Y, "o", 3)

f(x, y)

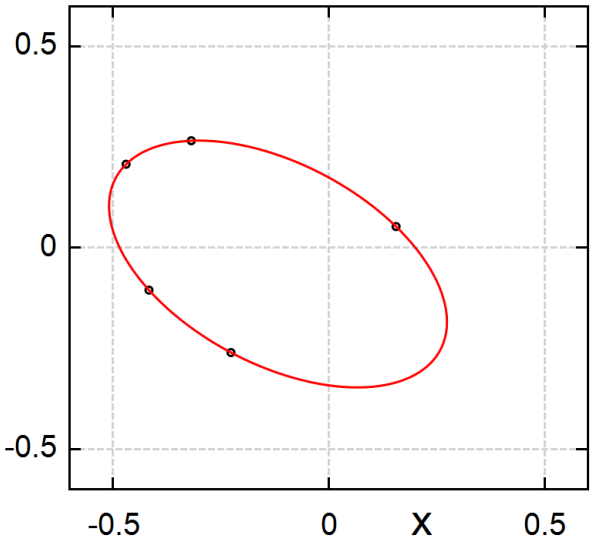


Рис. 11. Рисуем гиперболу или эллипс по пяти точкам

Можно нажимать клавишу F9 (пересчет – бросание камешков в воду) и смотреть на… круги, пардон, эллипсы и гиперболы, таким нажатием образуемые. А чтобы это занятие не было пустою забавою, то можно подсчитывать, как часто получаются эллипсы или гиперболы. Из графика этого понять не всегда возможно: что это – дуга эллипса или дуга ветви параболы. Расчеты, показанные на рис. 11 и 12, помогут сделать это. По ним через решение системы пяти линейных алгебраических уравнений рассчитываются коэффициенты уравнения кривой второго порядка (рис. 11), а затем высчитывается значение инварианты D, по которой кривая идентифицируется не на глазок, а точно (рис. 12).

$$f(x, y) := a_{11} \cdot x^2 + 2 \cdot a_{12} \cdot x \cdot y + a_{22} \cdot y^2 + 2 \cdot a_{13} \cdot x + 2 \cdot a_{23} \cdot y + a_{33}$$

$$a_{33} := 1$$

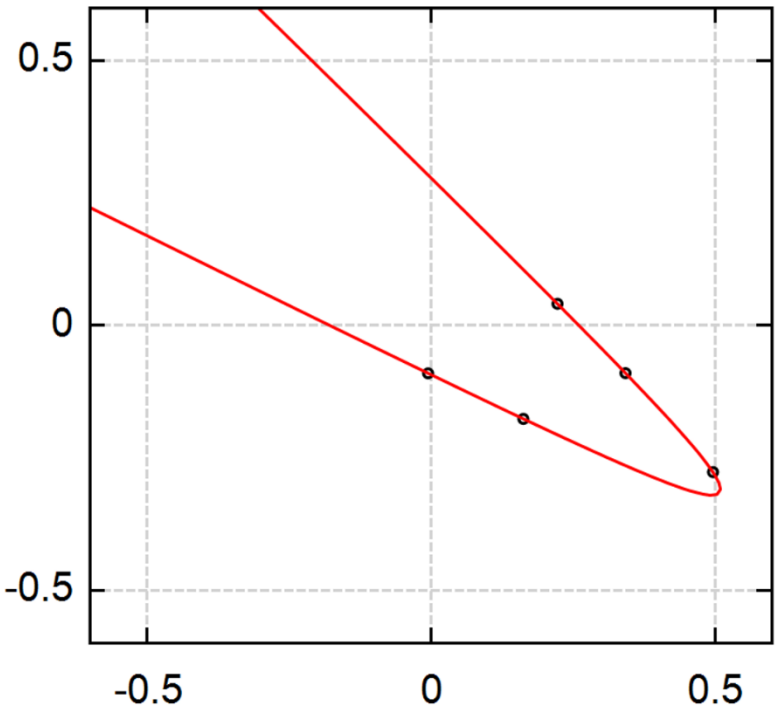
$$M := \begin{bmatrix} X_1^2 & 2 \cdot X_1 \cdot Y_1 & Y_1^2 & 2 \cdot X_1 & 2 \cdot Y_1 \\ X_2^2 & 2 \cdot X_2 \cdot Y_2 & Y_2^2 & 2 \cdot X_2 & 2 \cdot Y_2 \\ X_3^2 & 2 \cdot X_3 \cdot Y_3 & Y_3^2 & 2 \cdot X_3 & 2 \cdot Y_3 \\ X_4^2 & 2 \cdot X_4 \cdot Y_4 & Y_4^2 & 2 \cdot X_4 & 2 \cdot Y_4 \\ X_5^2 & 2 \cdot X_5 \cdot Y_5 & Y_5^2 & 2 \cdot X_5 & 2 \cdot Y_5 \end{bmatrix} \quad v := \begin{bmatrix} -a_{33} \\ -a_{33} \\ -a_{33} \\ -a_{33} \\ -a_{33} \end{bmatrix}$$

$$|M| = 0.00214$$

$$\begin{bmatrix} a_{11} \\ a_{12} \\ a_{22} \\ a_{13} \\ a_{23} \end{bmatrix} := M^{-1} \cdot v = \begin{bmatrix} 17.69 \\ -0.1647 \\ -1.25 \\ 4.224 \\ 0.1226 \end{bmatrix}$$

Рис. 11. Расчёт коэффициентов уравнения кривой второго порядка

Кстати, дугу эллипса от дуги гиперболы визуально можно отличить и так – изменить масштаб графика, чтобы увидеть вторую ветвь гиперболы – сравните графики на рис. 12.



$$D := \begin{vmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{vmatrix} = -96.96$$

$$\begin{cases} \text{"Эллипс"} & \text{if} \quad D > 0 \\ \text{"Гипербола"} & \text{if} \quad D < 0 \\ \text{"Парабола"} & \text{otherwise} \end{cases} = \text{"Гипербола"}$$

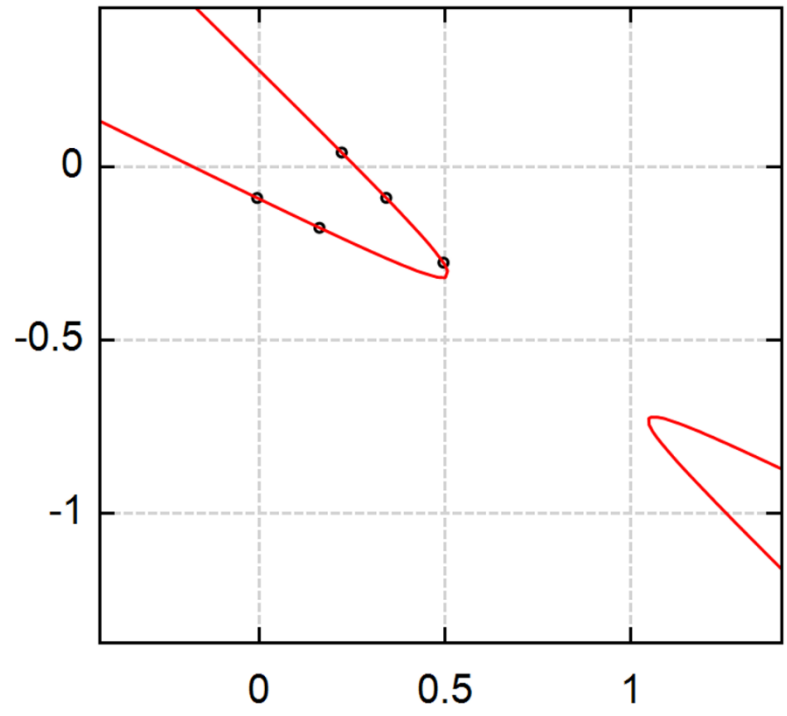


Рис. 12. Эллипс или гипербола?

Гиперболы на графиках получаются разные. В одном случае три точки находятся на одной ветви гиперболы, а две – на другой (см. рис. 11), а в другом случае (рис. 12) все пять точек оказываются на одной ветви гиперболы. Эти случаи тоже можно подсчитывать, бросая камешки в воду и задумываясь о возможной новой математической константе. О новуой математической задаче коммивояжёра, следующим по кубической параболе, мы уже задумались.

В операторе выбора в середине на рис. 12 перечислены эллипс, гипербола и парабола, которая практически никогда не выпадет при пяти точках. Тут вспоминается старый академический анекдот. Студент просыпается по будильнику, подбрасывает камешек, пардон, монетку и загадывает: «Выпадет орел – повернусь на правый бок, решка – на левый, а станет на ребро – пойду на лекцию!». Так вот, проведение параболы через пять точек – это падение монетки на ребро или даже зависание её в воздухе. А сколько нужно точек для параболы?

Если задать этот вопрос не только школьникам или студентам, но и более зрелым и математически подкованным людям, то 90% ответят – три, 9% – пять. Первый ответ связан со школьной параболой, ось которой параллельна оси ординат, а второй с тем, что для кривой второго порядка (в том числе и для параболы) требуется пять точек – см. выше. Но правильный ответ (100 – 90 – 9 = 1%), как это часто бывает, находится посередине – нужно иметь четыре точки, через которые проводятся не одна, а две параболы. На рисунке 13 показан соответствующий расчёт с системой четырех уже не линейных, а нелинейных уравнений. Система нелинейных уравнений в отличие от системы линейных уравнений (рис. 12) требует для решения начального приближения, что и определяет численный поиск одного корня из двух через встроенную в SMath функцию `roots`. Нюансы этих вычислений обсуждались на форуме пользователей SMath по адресу https://en.smath.com/forum/yaf_poststst25423_Roots---4-equations.aspx.

$$n := 5 \qquad X := \text{Random}(n) \qquad Y := \text{Random}(n)$$

$$f(x, y) := A \cdot x^2 + B \cdot x \cdot y + C \cdot y^2 + D \cdot x + E \cdot y + F \qquad B^2 - 4 \cdot A \cdot C = 0 \qquad F := 1$$

$$\begin{bmatrix} B \\ C \\ D \\ E \end{bmatrix} := \text{roots}\left(\left[\begin{array}{l} \frac{B^2}{4 \cdot C} \cdot X_1 + B \cdot X_1 \cdot Y_1 + C \cdot Y_1^2 + D \cdot X_1 + E \cdot Y_1 + F = 0 \\ \frac{B^2}{4 \cdot C} \cdot X_2 + B \cdot X_2 \cdot Y_2 + C \cdot Y_2^2 + D \cdot X_2 + E \cdot Y_2 + F = 0 \\ \frac{B^2}{4 \cdot C} \cdot X_3 + B \cdot X_3 \cdot Y_3 + C \cdot Y_3^2 + D \cdot X_3 + E \cdot Y_3 + F = 0 \\ \frac{B^2}{4 \cdot C} \cdot X_4 + B \cdot X_4 \cdot Y_4 + C \cdot Y_4^2 + D \cdot X_4 + E \cdot Y_4 + F = 0 \end{array}\right], \begin{bmatrix} B \\ C \\ D \\ E \end{bmatrix}, \right)$$

Первое предположение

$$\begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} B \\ C \\ D \\ E \end{bmatrix} = \begin{bmatrix} 2.02381 \\ 1.27658 \\ -2.42926 \\ -2.18071 \end{bmatrix} \qquad A := \frac{B^2}{4 \cdot C} = 0.8021$$

$$\begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} B \\ C \\ D \\ E \end{bmatrix} := \begin{bmatrix} -113.42 \\ 41.694 \\ 17.086 \\ -26.806 \end{bmatrix} \qquad A := \frac{B^2}{4 \cdot C} = 77.13$$

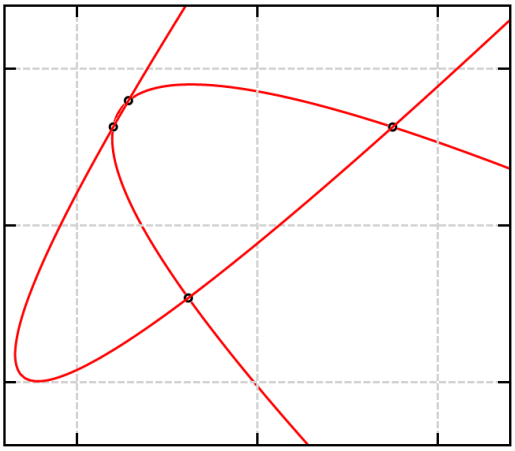


Рис. 13. Четыре случайные точки и две параболы

В заключении для затравки мы помещаем рис. 14, где показано, как через 1325 точек проведена кривая 50-го порядка! Ссылки в статьях [3, 4] помогут понять, как рисовался такой клубок ниток квадратной формы!

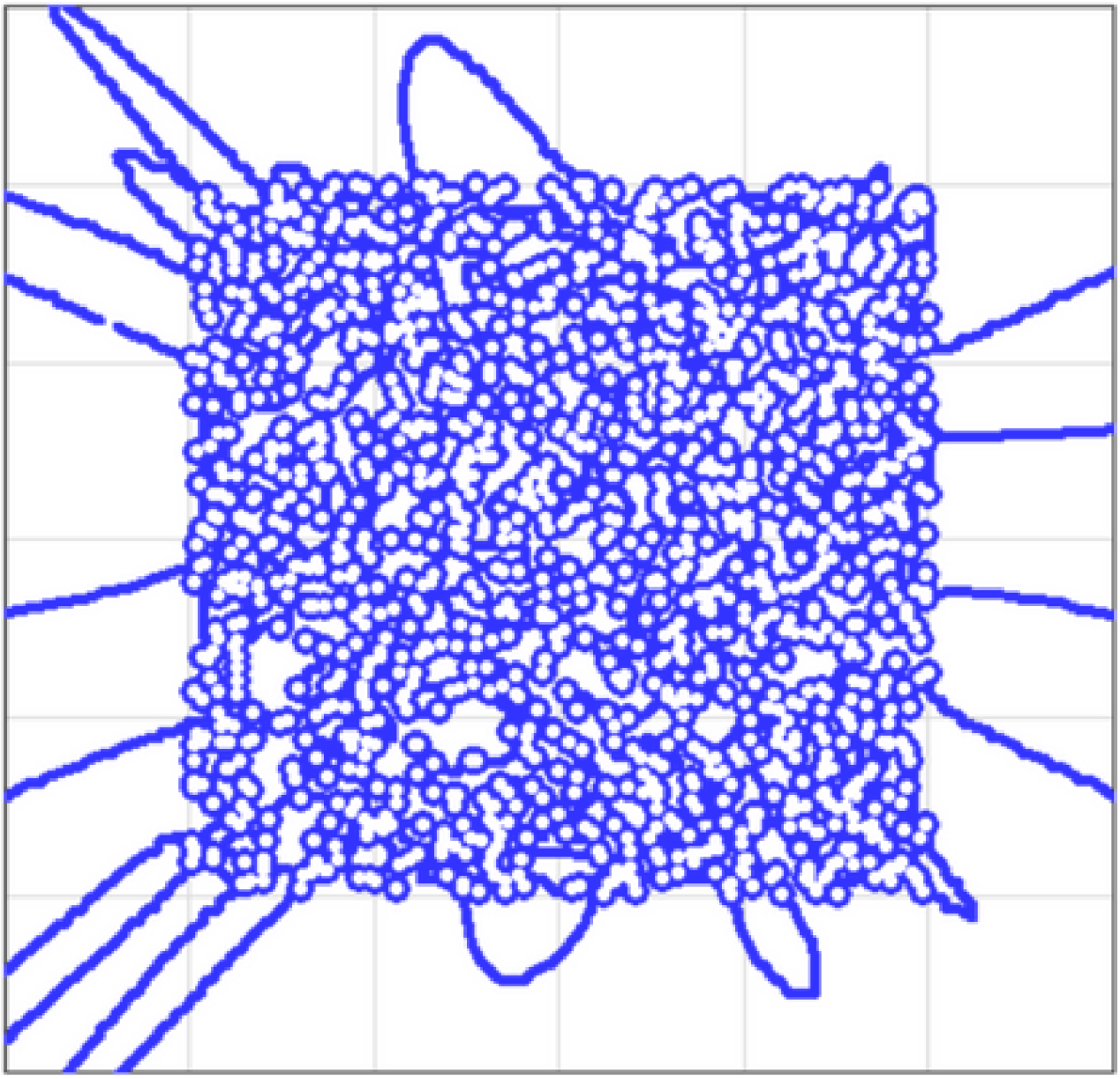


Рис. 14. Кривая 50-го порядка, проведённая через 1325 точек

Выводы

Математические эксперименты с компьютером могут вылиться в увлекательное занятие с вероятностью создания новых задач и новых математических констант.

Литература:

1. Информационные технологии в инженерных расчетах: SMath и Python / В. Ф. Очков, К. А. Орлов, Ю. В. Чудова [и др.]. — Санкт-Петербург : Лань, 2023. — 212 с. — ISBN 978-5-507-45821-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/319406 (дата обращения: 05.10.2024). — Режим доступа: для авториз. пользователей.

2. https://ru.wikipedia.org/wiki/Задача_коммивояжёра
3. Очков В. Ф., Елисеев А. Г., Федоров Ю. С. Статистическая задача об эллипсе и гиперболе или Новогодняя математическая открытка // Cloud of Science. № 4. 2018. С. 592-598 (http://twt.mpei.ac.ru/ochkov/Hyperbola-Ellipse.pdf)
4. Valery Ochkov. New Year Mathematical Card or V Points Mathematical Constant // Recreational Mathematics Magazine, Number 11, 2019, 27-33 pp (https://doi.org/10.2478/rmm-2019-0003)