1. Лабораторная работа №3

Решение задач комбинаторной оптимизации с помощью генетических алгоритмов на примере задачи укладки рюкзака

Цель работы: Решение задач комбинаторной оптимизации с помощью генетических алгоритмов на примере задачи укладки рюкзака.

При выполнении лабораторной работы можно использовать следующие источники из прилагаемого списка литературы [1,3,8-10].

Задачи комбинаторной (дискретной) оптимизации сильно отличаются от задач непрерывной оптимизации, которые рассматривали до сих пор. В численной («непрерывной») оптимизации большое значение имеет выбор направления поиска решения в пространстве решений. Например, градиентные и близкие методы , в значительной мере используют эту информацию. При этом неявно используется «близость» решений в пространстве поиска. В комбинаторной оптимизации близость решений сильно зависит от кодирования решений и направления поиска фактически нет. Это сильно осложняет решений таких задач. Далее рассматривается на примере задачи укладки рюкзака применение генетических алгоритмов для задач комбинаторной оптимизации.

Задача об укладке рюкзака

Эта задача имеет следующую неформальную постановку. Имеется рюкзак объемом W и n различных предметов. Каждый предмет i имеет известный объем W_i и стоимость P_i ($i = 1 \ldots n$). В рюкзак можно положить целое число различных предметов. Нужно упаковать рюкзак так, чтобы полная стоимость уложенных предметов была максимальной, а их общий объем не превышал заданный объем W – емкость рюкзака. Форма предметов здесь не учитывается .

Пример : Рюкзак объемом W=90

Таблица 3.1 Пример 3.1 данных для задачи укладки рюкзака

Item	1	2	3	4	5	6	7
p_i	6	5	8	9	6	7	3
w_i	2	3	6	7	5	9	4
p_i/w_i	3	1.67	1.33	1.29	1.2	0.78	0.75

Постановка задачи: необходимо найти вариант укладки рюкзака, который дает максимум стоимости уложенных предметов при заданном ограничении на объем (рюкзака), т.е.

$$f(\mathbf{x}) = \max \sum_{i=1}^{n} p_i x_i \qquad \sum_{i=1}^{n} w_i x_i \le W$$

$$x_i \in \{0,1\}, \; i = 1,2,\cdots,n$$

Т.е. для данного множества весов W_i , стоимостей P_i и объема W надо найти двоичный вектор X= ($x_1,\ldots, x_n$), где $x_i=1$, если предмет укладывается в рюкзак;$x_i=0$, если предмет не укладывается; при этом должно выполняться: $V = \sum_{i=1}^{n} W_i \le W$, и $\sum P_i x_i = \max$.

Рассмотрим различные варианты кодирования потенциальных решений для этой задачи на нашем примере. «Естественным» является эвристический («жадный») подход, при котором в первую очередь укладываются предметы с большей стоимостью и меньшим весом. Поэтому можно вычислить отношения p_i /w_i для всех предметов , упорядочить (ранжировать) предметы в порядке убывания и производить укладку в соответствии с этим порядком с проверкой ограничения (по весу или объему). То есть первым укладываем предмет с наибольшим значением p_i /w_i , затем следующий по порядку предмет и т.д. При этом необходимо на каждом шаге проверять ограничение по весу(объему). Для нашего примера решением в соответствии с этим алгоритмом является укладка {1,2,7}, при этом стоимость предметов P= 6+5+7=18 и вес W=2+3+4=9. Рассмотрим несколько методов кодирования потенциальных решений для этой задачи.

1. **Двоичный код**

Здесь каждому i-му предмету соответствует булева переменная x_i и 1 разряд двоичного кода. При этом (как указано выше) значение $x_i=1$, если предмет укладывается в рюкзак;$x_i=0$, если предмет не укладывается.

1	2	3	4	5	6	7
0	1	0	1	0	1	0

Например, двоичный код представляет решение, где укладываются предметы 2, 4,6. Заметим, что такое кодирование может давать неправильные решения (с превышением веса(объема)).

В качестве фитнесс-функции в простейшем случае можно взять $P(X) = \sum_{i=1}^{n} x_i \cdot P_i$, но в этом случае, как указано выше, есть проблемы с неправильными решениями. Дело осложняется тем, что хорошие решения, естественно, лежат близко границе допустимых решений.

2. Кодирование переменной длины

В этом случае длина хромосомы может изменяться в процессе искусственной эволюции. Здесь положение (локус) гена значения не имеет, а значение гена (аллель) определяет порядок предмета. Например, (1,6,5) означает укладку в рюкзак предметов 1, 6 и 5. Код переменной длины предусматривает специальные методы сохранения допустимости решений. При инициализации популяции можно случайно генерировать n случайных перестановок (номеров предметов). Далее в порядке, определяемом, данной перестановкой мы проверяем каждый предмет на его укладку в рюкзак. Если укладка предмета не вызывает переполнения рюкзака, то рассматриваемы предмет укладывается в рюкзак. В противном случае данный предмет не укладывается и мы переходим к следующему предмету по порядку (в текущей перестановке). В этом случае одна случайная перестановка может породить 1 допустимое решение. Эта процедура повторяется для каждой перестановки.

При выполнении кроссинговера необходимо поддерживать правильность (допустимость) решения. То есть допустимые родительские особи должны порождать допустимые особи -потомки.

Рассмотрим эту проблему на последнем примере рюкзака с n=7 и W=90 , представленном табл.3.2. Здесь кроссинговер может выполняться следующим образом.

1. Выбрать случайную точку в 1-м родителе.
2. Выбрать случайный фрагмент во втором родителе.
3. Вставить выбранный фрагмент второго родителя в выбранную позицию первого родителя, удалить повторяющиеся предметы. При этом возможно построение недопустимых решений.
4. Для недопустимых решений выполнить процедуру восстановления на основе ранжирования предметов по отношениям p_i / w_i .

Следующий рис.3.4 иллюстрирует выполнение для указанного примера.

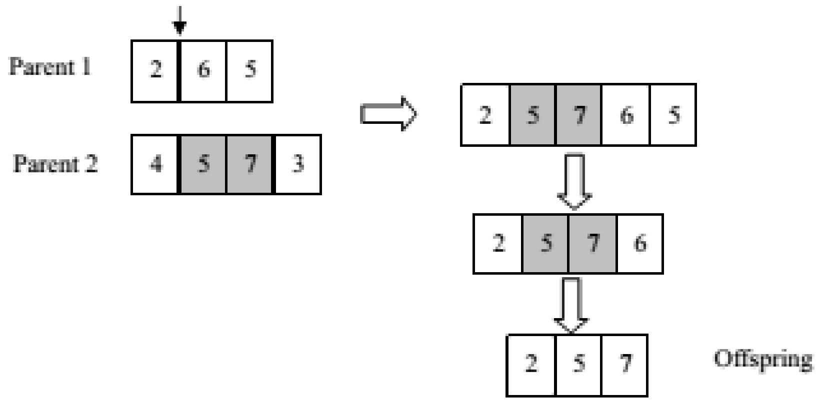

Рис.3.4 Пример выполнения оператора кроссинговера

Здесь после выполнения 3-го шага получено недопустимое решение (2,5,7,6). Согласно порядку предметов по отношению p_i /w_i представленной таблицы мы пытаемся Сначала удалить предмет 5. Но этого недостаточно – получаемое решение (2,7,6) остается недопустимым. Поэтому мы пытаемся удалить следующий по порядку предмет (из начальной конфигурации) и получаем допустимое решение (2,5,7).

Оператор мутации выполняется проще – можно удалить случайно выбранный предмет и на его место вставить случайно выбранный из оставшихся. Затем, в случае недопустимости построенного решения, выполнить шаг 4 предыдущего алгоритма.

3.Перестановочное кодирование

Любая перестановка n чисел может быть интерпретирована как последовательность укладываемых предметов. Слегка модифицируем последний пример: заменим на W=100 при сохранении n=7. Для особи (1,6,4,7,3,2,5) декодирование можно выполнить следующим образом. Мы укладываем в рюкзак первый предмет из этой перестановки –предмет 1. Далее укладываем второй предмет – предмет 6 и это не вызывает переполнения поскольку 40+40=80<100=W. Затем аналогично поступаем с предметом 4, который также не вызывает переполнения -40+40+10=90<100=W.
Но при попытке уложить следующий согласно данной перестановке предмет 7 имеет место переполнение - 40+40+10+30=120>100=W. Поэтому предмет 7 не укладывается. Аналогично мы вынуждены пропустить предметы 3 и 2, поскольку они также вызывают переполнение. Наконец, последний предмет 5 не вызывает переполнения - 40+40+10+10=10=100=W. В результате декодирования перестановки (1,6,4,7,3,2,5) получаем особь – решение задачи (1,6,4,5).

Таким способом любая перестановка декодируется в допустимое решение. Заметим, что обычный кроссинговер и мутация могут нарушить

допустимость решения. Данный метод декодирования порождает отображение много-к-одному.
Например, перестановки (1,6,4,7,3,2,5), (1,4,6,7,3,2,5), (6,4,1,7,3,2,5), …,(6,4,1,7,3,5,2) при декодировании порождают одно и тоже решение – (1,6,4,5). Это снижает эффективность кодирования за счет повышения размерности пространства поиска. Так для n предметов при двоичном кодировании пространство поиска содержит 2^n вариантов решений. Для перестановочного кода пространство решений –n! , которое с ростом n растет гораздо быстрее, что показывает следующая табл.3.4.

Таблица 3.4. Сложность пространства решений

n	1	2	3	4	5	6	7	8	9	10
2^n	2	4	8	16	32	64	128	256	512	1024
$n!$	1	2	6	24	120	720	5040	40320	362880	3628800

Таким образом сохранение допустимости решений достигается в этом случае ценой расширения пространства поиска (и снижения эффективности).

Данная задача относится к классу задач с ограничениями, при решении которых применяются следующие подходы :

1) введение в фитнесс-функцию дополнительного штрафа;

2) использование алгоритмов “восстановления” некорректных решений.

1) В первом случае в фитнесс-функцию вводится дополнительная штрафная функция, которая для неправильных решений дает большие отрицательные значения ЦФ.

При этом задача с ограничениями трансформируется в задачу без ограничений путем назначения штрафа для некорректных решений.

Фитнесс-функция для каждой особи может быть определена следующим образом

$f(X)=\sum_{i=1}^{n} x_i \cdot P_i - Pen(X)$.Разработано множество методов назначения штрафных значений, например:

а) $Pen(X)=\log_2(1+\rho\cdot(\sum_{i=1}^{n} x_i \cdot W_i - W))$,

б) $Pen(X)=\rho\cdot(\sum_{i=1}^{n} x_i \cdot W_i - W)$, в) $Pen(X) = (\rho \cdot (\sum_{i=1}^{n} x_i \cdot W_i - C))^2$.

Здесь для всех трех случаев $\rho = \max_{1 \le i \le n} \{P_i / w_i\}$.

Кроме этого, иногда выполняют масштабирование согласно формуле

$$\delta = \min\left\{W, \left|\sum_{i=1}^{n} w_i - W\right|\right\},$$

и применяется следующий вид штрафной функции:

$$p(\mathbf{x}) = 1 - \frac{\left|\sum_{i=1}^{n} w_i x_i - W\right|}{\delta}$$

Графическая интерпретация этой функции представлена на следующем рис.3.1.

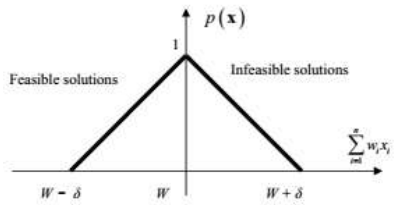

Рис.3.1 Штрафная функция

Фитнесс-функцию в этом случае можно определить так $fitness(\mathbf{x}) = f(\mathbf{x})\, p(\mathbf{x})$

2) Второй подход к решению задач с ограничениями основан на специальных алгоритмах “восстановления” некорректных решений.

Рассмотрим на следующем примере с W=90 и n=7, параметры предметов которого представлены в следующей табл.3.1.

Таблица 3.2 Пример 3.2 данных для задачи укладки рюкзака

Item	1	2	3	4	5	6	7
p_i	40	60	10	10	3	20	60
w_i	40	50	30	10	10	40	30
p_i/w_i	1	1.2	0.33	1	0.33	0.5	2

.

1	2	3	4	5	6	7
0	1	0	1	0	1	0

В этом случае решение (см. выше), очевидно, является неправильным (недопустимым), поскольку сумма весов $w_2 + w_4 + w_6 = 50+10+40=100 > 90=W$. В соответствии с табл.3.2 можно удалить предмет 6 с минимальным отношением p_i / w_i и это делает решение допустимым.

Следует отметить, что многие алгоритмы восстановления требуют значительных вычислительных ресурсов, и полученные решения иногда требуют адаптации к конкретным практическим приложениям. В этом случае в качестве фитнесс-функции используется $f(X') = \sum_{i=1}^{n} x_i' P_i$, где вектор X' – восстановленная версия исходного вектора X. Здесь следует отметить, по крайней мере, два аспекта.

Во-первых, можно использовать различные алгоритмы восстановления.

Во-вторых, восстановленные особи могут замещать только некоторую часть исходных особей в популяции.

Процент замещаемых особей может варьироваться от 0% до 100% и его значение является важнейшим параметром метода восстановления. В некоторых работах отмечается, что наилучшие результаты получаются при 5%, во всяком случае, лучше, чем в двух крайних случаях – 0% (без замещения) и 100% (любая восстановленная особь заменяет исходную).

Ниже приведен простой алгоритм восстановления.

```
Восстановление(X)
 {
  переполнение_рюкзака=false;
  X'=X;
  if ( f(X') = Σ_{i=1}^{n} x_i'·W_i > C )
     then переполнение_рюкзака=true;
 while(переполнение_рюкзака)
        {
          i=выбор предмета из рюкзака;
          // удаление выбранного предмета из рюкзака;
            x'_i = 0;
   if ( f(X') = Σ_{i=1}^{n} x'_i·W_i ≤ C )
            then переполнение_рюкзака=false;
          }     }
```

При этом используются два основных способа выбора объекта:

а) случайный выбор объекта из рюкзака;

б) “жадное восстановление”, при котором вначале все предметы сортируются в порядке убывания их стоимости P_i и на каждом шаге для удаления выбирается предмет минимальной стоимости (из имеющихся в рюкзаке).

3) Третий подход к решению задач с ограничениями использует специальное отображение (декодирование) особей, которое гарантирует генерацию допустимого решения (с учетом ограничений), или используют

проблемно-ориентированные генетические операторы, сохраняющие корректность решения.

3.2 Практическая часть

Задание

1. Реализовать с использованием генетических алгоритмов решение задачи укладки рюкзака простой сложности по индивидуальному заданию согласно номеру варианта (см. таблицу 1) использованием соответствующего кодирования.
2. Сравнить найденное решение с представленным в условии задачи оптимальным решением.
3. Реализовать с использованием генетических алгоритмов решение задачи укладки рюкзака повышенной сложности по индивидуальному заданию согласно номеру варианта соответствующего кодирования (см. таблицу 2).
4. Представить графики сходимости решения с различными параметрами генетического алгоритма: мощности популяции, значений вероятностей различных видов скрещивания, мутации.
5. Проанализировать время выполнения и точность нахождения результата в зависимости от значений вероятностей различных видов скрещивания, мутации.

Содержание отчета.

1. Титульный лист.
2. Индивидуальное задание по варианту.
3. Краткие теоретические сведения.
4. Программа и результаты выполнения индивидуального задания с комментариями и выводами.
5. Ответ на контрольный вопрос.

Таблица 1. Варианты заданий

Вариант	Номер тестовых данных задания1 простой сложности	Номер тестовых данных задания 2 повышенной сложности	Вид кодирования решения	Контрольный вопрос
1	1	10	двоичное	1
2	2	9	переменной длины	2
3	3	8	перестановочное	3
4	4	7	двоичное	4
5	5	6	переменной длины	5
6	6	5	перестановочное	6
7	7	4	двоичное	7
8	1	3	переменной длины	8
9	2	2	перестановочное	9
10	3	1	двоичное	10
11	4	10	переменной длины	1
12	5	9	перестановочное	2
13	6	8	двоичное	3
14	7	7	переменной длины	4
15	1	6	перестановочное	5
16	2	5	двоичное	6
17	3	4	переменной длины	7
18	4	3	перестановочное	8
19	5	2	двоичное	9

20	6	1	переменной длины	10
21	7	10	перестановочное	1
22	1	9	двоичное	2
23	2	8	переменной длины	3
24	3	7	перестановочное	4
25	4	6	двоичное	5
26	5	5	переменной длины	6
27	6	4	перестановочное	7
28	7	3	двоичное	8
29	1	2	переменной длины	9
30	2	1	перестановочное	10

Контрольные вопросы

1. При решении каких задач комбинаторной оптимизации может быть использован простой ГА с двоичным кодированием хромосом?
2. Какие модификации необходимы для эффективного использования простого ГА для решения задачи укладки рюкзака?
3. Какие виды штрафных функций могут быть использованы в фитнесс-функции при решении задачи укладки рюкзака?
4. В чем суть алгоритма восстановления при решения задачи укладки рюкзака?
5. Поясните понятие пространства решений на примере задачи укладки рюкзака.
6. В чем основная идея применения ГА для решения задачи укладки рюкзака?
7. Опишите структуру ГА для решения задачи укладки рюкзака.
8. Опишите двоичное кодирование для задачи укладки рюкзака.
9. Опишите кодирование переменной длины для задачи укладки рюкзака.
10. Опишите перестановочное кодирование для задачи укладки рюкзака.

Приложение

Тестовые данные для задачи укладки рюкзака

(benchmarks)

Задание 1 (сложность - простая)

1.Тестовые данные с сайта

емкость

Формат

Сначала задается емкость рюкзака,

Затем веса укладываемых предметов,

Далее стоимость укладываемых предметов,

И оптимальное решение – список уложенных предметов.

P01 is a set of 10 weights and profits for a knapsack of capacity 165.

Набор 1

Емкость

```
165

Веса
   23
   31
   29
   44
   53
   38
   63
   85
   89
   82
Стоимость
   92
   57
   49
   68
   60
   43
   67
   84
   87
   72
Оптимум
1
1
1
1
0
1
0
0
0
```

0

P02 is a set of 5 weights and profits for a knapsack of capacity 26.
Набор 2
Емкость
26

веса
12
7
11
8
9
Стоимость
24
13
23
15
16
Оптимум
0
1
1
1
0

P03 is a set of 6 weights and profits for a knapsack of capacity 190. Набор 3

Емкость
190
Веса
56
59
80
64
75
17
Стоимость
50
50
64
46
50
5

Оптимум
1
1
0

0
1
0

P04 is a set of 7 weights and profits for a knapsack of capacity 50.
Набор 4

Емкость
50
Веса
1
1
0
0
1
0

Стоимость
70
20
39
37
 7
 5
10

Оптимум
1
0
0
1
0
0
0

P05 is a set of 8 weights and profits for a knapsack of capacity 104.
Набор 5
Емкость
104

Веса
25
35
45
 5
25
 3
 2
 2
 Стоимости
350

400
450
 20
 70
  8
  5
  5

Оптимум
1
0
1
1
1
0
1
1

P06 is a set of 7 weights and profits for a knapsack of capacity 170. The knapsack can be packed to an optimal weight of 169.
Набор 6
Емкость
170

Веса
41
50
49
59
55
57
60

Стоимости
442
525
511
593
546
564
617

P07 is a set of 24 weights and profits for a knapsack of capacity 6404180, from Kreher and Stinson, with an optimal profit of 13549094.
Набор 8
Емкость
6404180
 Веса
382745

799601
909247
729069
467902
44328
34610
698150
823460
903959
853665
551830
610856
670702
488960
951111
323046
446298
931161
31385
496951
264724
224916
169684
Стоимости
825594
1677009
1676628
1523970
943972
97426
69666
1296457
1679693
1902996
1844992
1049289
1252836
1319836
953277
2067538
675367
853655
1826027
65731
901489
577243
466257
369261

Оптимум
1
1
0

```
1
1
1
0
0
0
1
1
0
1
0
0
1
0
0
0
0
0
1
1
1
```

Задание 2 (сложность - повышенная)

***.Тестовые данные для задачи укладки рюкзака* kplib.**

Тестовые данные сгенерированы согласно следующему источнику.

Kellerer, H., Pferschy, U., & Pisinger, D. (2004). Exact solution of the knapsack problem. In Knapsack Problems (pp. 117-160). Springer, Berlin, Heidelberg.

Код генерации доступен здесь.

- https://bitbucket.org/likr/mnkproblems/
- https://bitbucket.org/likr/mnkptools/

Данные представлены в следующем формате:

```
n -  количество укладываемых предметов.
C   - объем рюкзака.

стоимость_предмета  объем_предмета
```

```
p_1 w_1
p_2 w_2
…
p_n w_n
```

Набор 1 некоррелированные

50 число предметов

14778 объем рюкзака

845 804 `стоимость 1-го предмета` `объем 1-го предмета`

758 448 `стоимость 2-го предмета` `объем 2-го предмета`

421 81

259 321

512 508

405 933

784 110

304 552

477 707

584 548

909 815

505 541

282 964

756 604

619 588

251 445

910 597

983 385

811 576

903 291

311 190

730 187

899 613

684 657

473 477

101 90

435 758

611 877

914 924

967 843

478 899

866 924

261 541

806 392

549 706

15 276

720 812

399 850

825 896

669 590

2 950

494 580

868 451

244 661

326 997

871 917

192 794

568 83

239 613

968 487

Набор 2 некоррелированные

50
13810

957 491
948 925
57 501
85 832
836 354
736 883
670 900
309 462
606 568
607 921
582 724
159 487
431 222
394 325
724 700
995 167
950 908
545 269
445 912
269 310
36 958
28 707
465 505
319 518
381 652
892 588
526 312
561 208
237 512
24 935

326 624
137 76
511 821
999 726
675 908
182 192
894 745
797 59
735 653
907 274
763 227
790 876
354 107
981 523
962 854
162 245
755 211
716 881
462 423
531 717

Набор 3 некоррелированные

50
13596

238 682
545 929
370 857
604 991
626 672
66 164
14 861
838 965
260 905
235 570
996 714
471 212
837 832
477 574
640 285
151 64
635 854
869 990
524 89
742 801
672 411
65 151
759 294
592 769
302 873
32 45
866 615
473 45
719 719
879 331
715 881
922 981
395 506
801 999
445 310
936 77

879 600
98 32
136 198
217 408
966 611
437 157
627 43
302 868
508 314
386 959
351 897
586 378
585 461
905 521

Набор 4 некоррелированные

50
11618

237 281
104 535
397 472
155 343
67 998
402 196
918 413
801 203
766 633
222 277
537 356
277 747
173 321
107 559
215 905
928 101
829 62
807 229
801 766
194 616
310 238
627 332
732 178
855 460
881 43
87 698
606 896
672 955
506 735
178 960
474 19
90 289
935 967
866 776
548 411
301 944
909 621
573 818
883 294
849 192
509 445
414 137

599 382
432 962
162 332
306 10
813 45
44 170
47 784
627 363

Набор 5 некоррелированные

50
11922

623 338
742 310
796 819
943 481
740 316
923 482
30 705
466 58
944 976
649 23
901 750
114 845
470 19
247 788
544 367
574 579
14 10
217 47
280 181
917 956
766 197
160 756
798 930
139 943
618 345
127 355
2 525
872 776
210 109
216 749
983 798
873 860
290 37
962 946
540 92
678 341
205 611
941 919
691 340
967 925
894 546
299 313
362 317
166 178
146 79
66 149
302 690
604 997

4 162
678 49

Набор 6 некоррелированные

50
12044

794 881
822 526
486 117
262 663
1 313
663 197
471 484
760 139
374 210
771 866
273 699
802 13
730 779
415 18
539 333
683 905
193 621
554 307
806 378
266 390
804 127
686 1000
845 54
336 423
94 747
801 398
805 981
446 246
94 980
198 893
635 671
292 890
952 448
589 832
201 797
656 517
361 471
933 706
910 410
515 86
645 747
698 266
806 488
977 785
29 173
362 22
602 124
305 590
590 362
90 257

Набор 7 некоррелированные

50

12828

324 981
151 119
651 419
73 758
536 152
366 489
58 40
508 669
38 765
434 574
70 876
91 314
425 696
827 595
124 580
224 457
628 840
948 945
578 475
397 665
977 61
47 702
859 648
290 994
145 822
118 285
309 386
817 669
181 23
582 462
639 169
373 118
548 59
63 769
60 130
206 248
681 391
428 872
315 81
586 450
454 550
300 884
795 820
699 864
245 279
575 416
526 359
876 885
730 958
288 151

Набор 8 некоррелированные

50
10985

227 703
963 950
127 844
705 504

86 198
248 151
1000 529
210 510
642 72
460 904
454 508
495 702
193 220
831 244
90 12
235 344
20 267
267 424
408 377
903 835
380 892
114 176
259 397
992 167
64 664
621 975
378 202
661 767
339 301
692 14
498 765
650 342
902 171
582 436
143 232
65 411
947 447
489 418
194 589
947 289
579 96
729 86
881 107
286 521
357 361
879 808
135 505
765 709
98 981
691 63

Набор 9 некоррелированные

50
11487

464 84
374 540
139 18
867 85
7 497
503 921
899 421
81 399
555 639
617 94

41 580
380 173
704 609
453 959
726 55
158 556
239 607
111 150
507 269
924 995
591 998
775 122
384 706
747 951
102 237
292 612
675 44
726 366
422 675
88 591
267 775
210 87
282 348
810 865
200 585
887 452
880 403
55 987
379 575
492 19
24 800
425 329
907 434
113 214
597 445
122 325
579 89
896 630
204 104
9 785

Набор 10 некоррелированные

50
11448

572 568
429 375
579 602
207 113
814 776
824 97
654 167
161 808
521 948
328 434
250 415
953 246
997 275
45 618
861 179
604 120

382 456
284 165
675 649
457 822
686 778
662 481
133 348
768 435
983 6
970 713
614 332
45 320
5 80
134 449
942 583
303 391
367 870
899 674
315 242
549 526
437 911
65 521
585 603
845 63
157 490
225 462
413 402
37 421
497 585
818 539
658 490
534 166
856 442
150 969