DoE gamification

Arnaud Legrand

2024-01-18

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
```

Test when setting up the shiny app

Using https://arnaud-legrand.shinyapps.io/design_of_experiments/?al1111 and injecting white noise (40 experiments) just to check that the shiny app was properly reset.

I did not take notes on how I did this but here is how to get it:

```
df=read.csv("Data",header=T)
str(df)
```

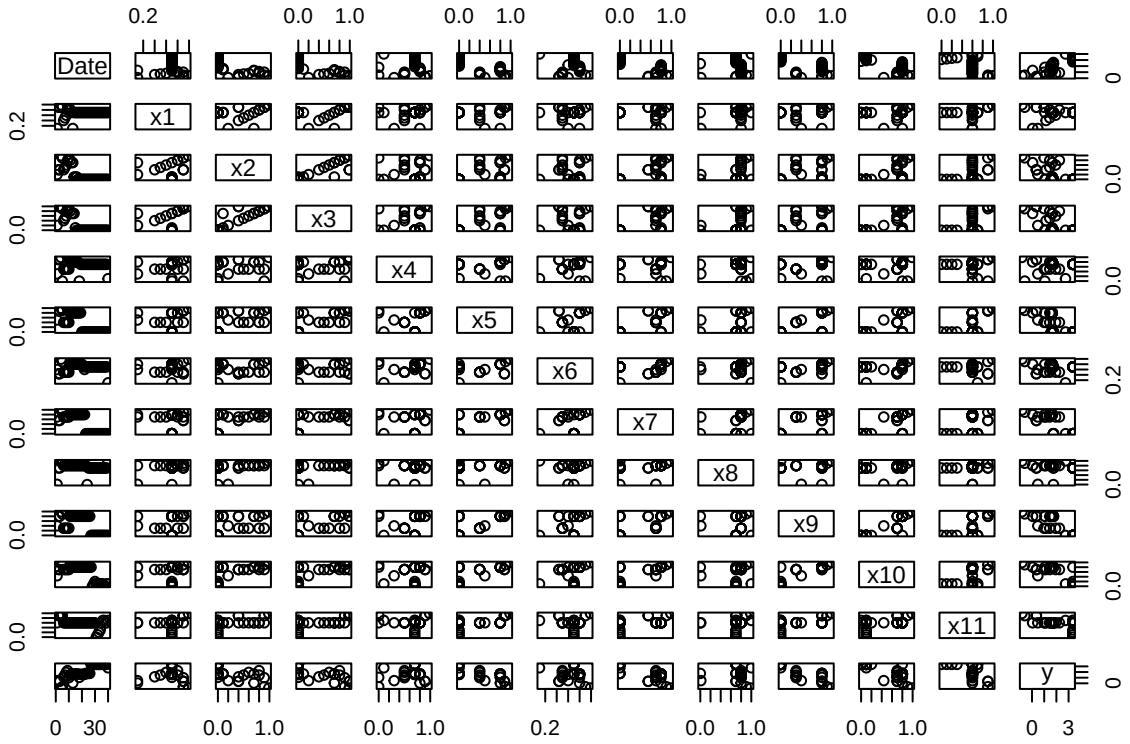
```
## 'data.frame':    66 obs. of  13 variables:
##  $ Date: chr  "2023-12-24-22:09:55" "2024-01-11-01:54:36" "2024-01-11-01:55:42" "2024-01-11-01:55:43
##  $ x1  : num  0.1 0.9 0.9 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x2  : num  0.2 0.4 0.4 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x3  : num  0.2 0.99 0.99 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x4  : num  0.3 0.9 0.9 0.99 0 0.5 0.5 0.5 0.5 0.5 ...
##  $ x5  : num  0.5 0.88 0.88 0.99 0.9 0.4 0.4 0.4 0.4 0.4 ...
##  $ x6  : num  0.6 0.44 0.44 0.99 0.9 0.5 0.5 0.5 0.5 0.5 ...
##  $ x7  : num  0.7 0.55 0.55 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x8  : num  0.01 0.7 0.7 0.99 0.9 0.8 0.8 0.8 0.8 0.8 ...
##  $ x9  : num  0.4 0.8 0.8 0.99 0.9 0.3 0.3 0.3 0.3 0.3 ...
##  $ x10 : num  0.44 0.7 0.7 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x11 : num  0.6 0.9 0.9 0.99 0.9 0.6 0.6 0.6 0.6 0.6 ...
##  $ y   : num  0.477 0.359 0.356 -0.828 -0.599 ...
```

```r
df = df[1:40,]
```

```r
summary(df)
```

```
##      Date                x1               x2               x3
##  Length:40          Min.   :0.1000   Min.   :0.0000   Min.   :0.0000
##  Class :character   1st Qu.:0.7000   1st Qu.:0.0000   1st Qu.:0.0000
##  Mode  :character   Median :0.7000   Median :0.0000   Median :0.0000
##                     Mean   :0.6873   Mean   :0.2147   Mean   :0.2567
##                     3rd Qu.:0.7000   3rd Qu.:0.4000   3rd Qu.:0.6250
##                     Max.   :0.9900   Max.   :0.9900   Max.   :0.9900
##        x4               x5               x6               x7
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.1000   Min.   :0.0000
##  1st Qu.:0.7000   1st Qu.:0.0000   1st Qu.:0.6750   1st Qu.:0.0000
##  Median :0.7000   Median :0.0000   Median :0.7000   Median :0.6250
##  Mean   :0.6498   Mean   :0.3337   Mean   :0.6793   Mean   :0.4198
##  3rd Qu.:0.7250   3rd Qu.:0.8000   3rd Qu.:0.8000   3rd Qu.:0.8000
##  Max.   :0.9900   Max.   :0.9900   Max.   :0.9900   Max.   :0.9900
##        x8              x9               x10              x11
##  Min.   :0.00   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.70   1st Qu.:0.0000   1st Qu.:0.1000   1st Qu.:0.6000
##  Median :0.80   Median :0.6000   Median :0.7000   Median :0.6000
##  Mean   :0.73   Mean   :0.4547   Mean   :0.5483   Mean   :0.5948
##  3rd Qu.:0.80   3rd Qu.:0.8000   3rd Qu.:0.8000   3rd Qu.:0.6000
##  Max.   :1.00   Max.   :0.9900   Max.   :0.9900   Max.   :0.9900
##        y
##  Min.   :-0.8279
##  1st Qu.: 1.4588
##  Median : 1.7095
##  Mean   : 1.9533
##  3rd Qu.: 3.3522
##  Max.   : 3.3569
```

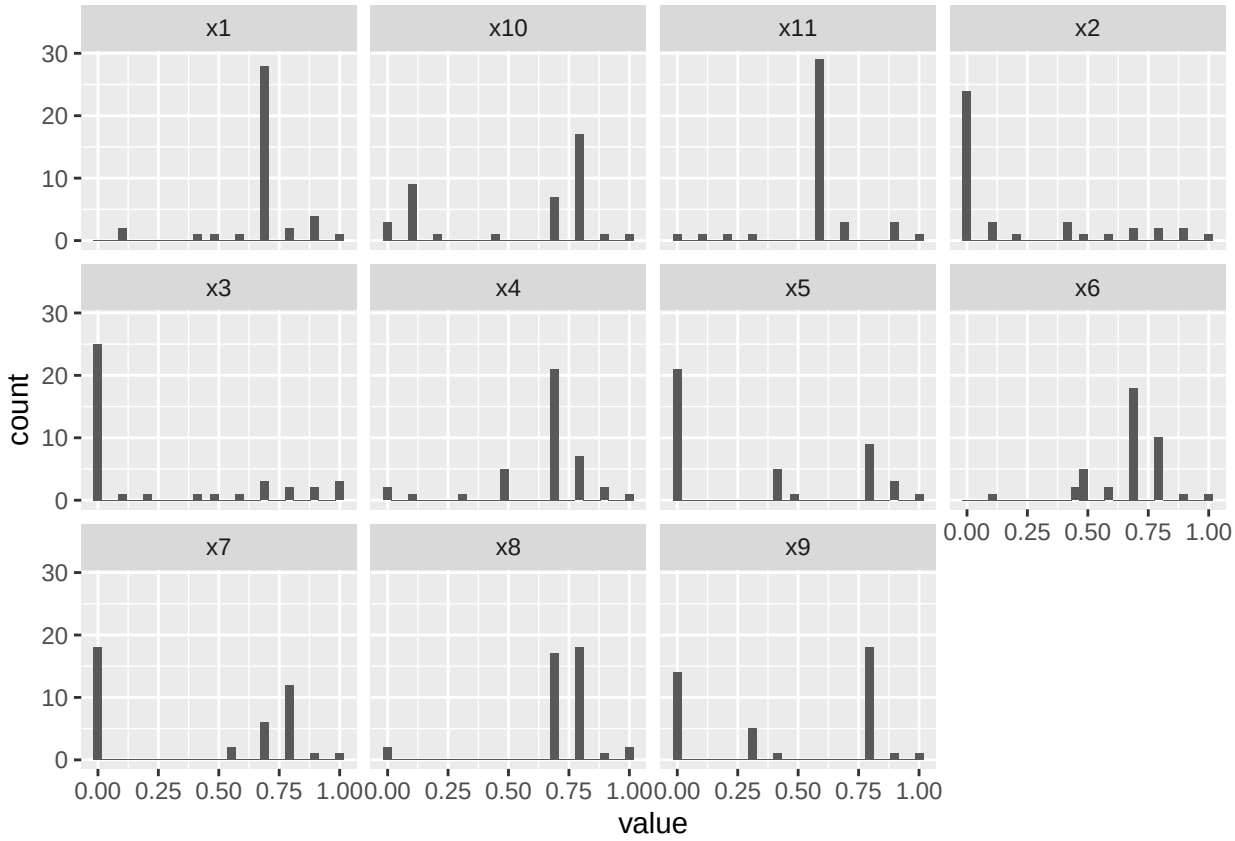
```r
plot(df)
```



Not so readable. Let's focus on inputs and check how they are spread.

```
df %>% select(-Date) %>% gather() %>% filter(key != "y") %>% ggplot(aes(x=value, group=key)) + geom_his
```

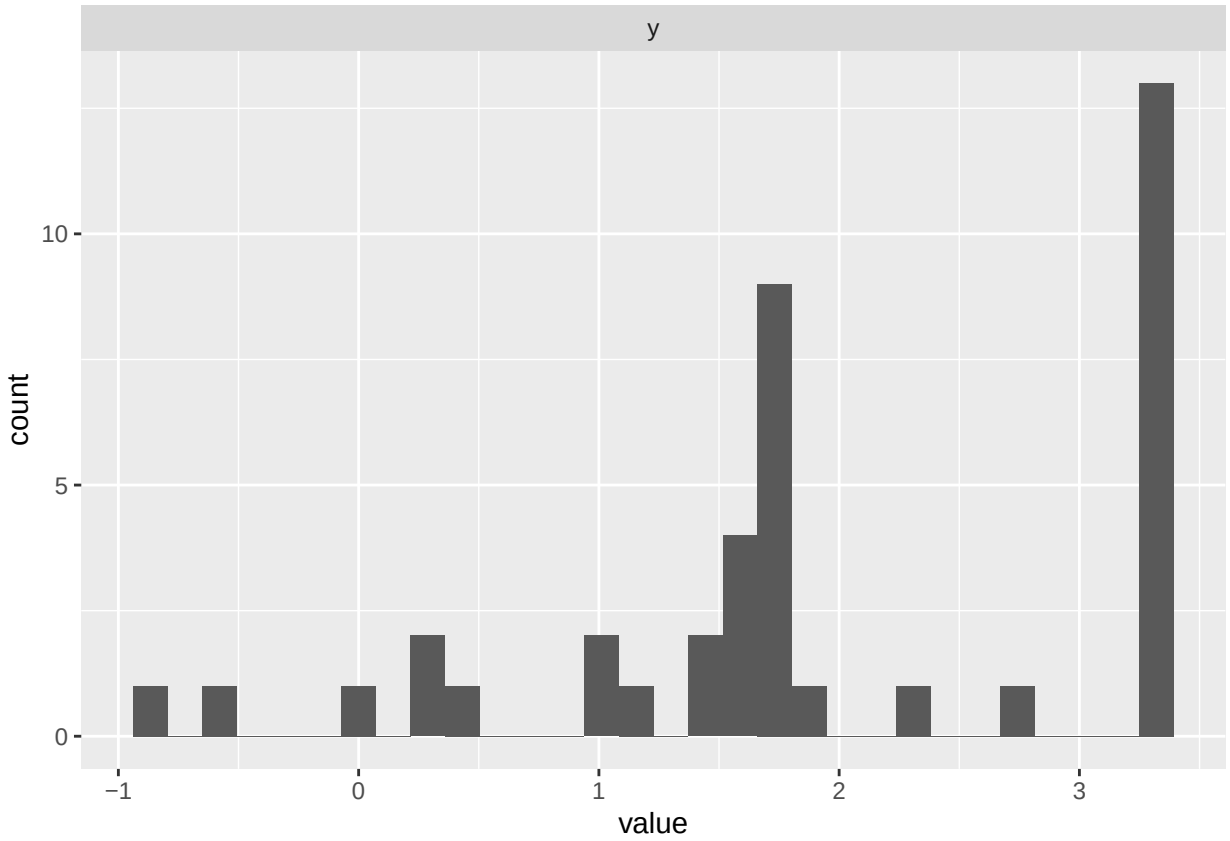
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Well, there are not so many points and uniformity appears like a reasonable assumption. What about the output now ?

```
df %>% select(-Date) %>% gather() %>% filter(key == "y") %>% ggplot(aes(x=value, group=key)) + geom_his
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



It's hard to tell what the influence of inputs may be but outputs appear to be distributed between -1 and 3 at the moment.

First interactions with the system

DoE

The most common assumption to all models we have seen is that $y = f(x) + \epsilon$, with ϵ being modeled as a random variable. Let's start by checking this and see how much knowledge we can get on ϵ.

I would like to evaluate the fluctuation of $f(x)$ around its mean for different values of x and test whether this fluctuation appears to depend on x or not. 10 replications seems the least I should do for such test. So I will generate a few combinations of x and I will generate these x in a uniform way to avoid bias. There is thus no need for sophisticated method such as a space-filling design. We're in dimension 11 and we cannot afford that much samples anyway.

```
set.seed(918682)
space_dim = 11
n_sample = 10
n_replicates = 10
x_init = runif(n=space_dim * n_sample, min=0, max=1)
df_init = as.data.frame(matrix(data=x_init, ncol = space_dim))
df_doe1 = data.frame()
for(i in 1:n_replicates) {
  df_doe1 = rbind(df_doe1,df_init)
```

```
}
df_doe1
```

```
##            V1         V2        V3         V4         V5         V6         V7
## 1   0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 2   0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 3   0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 4   0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 5   0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 6   0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 7   0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 8   0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 9   0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 10  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 11  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 12  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 13  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 14  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 15  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 16  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 17  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 18  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 19  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 20  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 21  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 22  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 23  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 24  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 25  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 26  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 27  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 28  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 29  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 30  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 31  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 32  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 33  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 34  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 35  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 36  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 37  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 38  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 39  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 40  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 41  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 42  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 43  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 44  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 45  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 46  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 47  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 48  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 49  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
```

```
## 50  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 51  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 52  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 53  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 54  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 55  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 56  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 57  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 58  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 59  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 60  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 61  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 62  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 63  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 64  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 65  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 66  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 67  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 68  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 69  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 70  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 71  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 72  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 73  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 74  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 75  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 76  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 77  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 78  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 79  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 80  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 81  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 82  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 83  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 84  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 85  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 86  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 87  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 88  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 89  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 90  0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
## 91  0.9626759 0.32483915 0.2590755 0.18594223 0.81913270 0.22815552 0.89156615
## 92  0.1076542 0.30467420 0.8847447 0.51341919 0.24594944 0.95125113 0.12925102
## 93  0.3084454 0.23235813 0.9365743 0.19361686 0.73418461 0.05219464 0.47697583
## 94  0.1187312 0.76511857 0.1931036 0.54999464 0.61089951 0.17716042 0.97643661
## 95  0.7394017 0.07501318 0.1344981 0.07748240 0.28677569 0.81739398 0.39639737
## 96  0.0584193 0.10780050 0.6259921 0.75585789 0.02574348 0.59986351 0.40277386
## 97  0.6968773 0.17475215 0.9529503 0.34869013 0.74491060 0.56139086 0.75369801
## 98  0.5879794 0.26163844 0.8077954 0.12571390 0.08703325 0.78898862 0.34966579
## 99  0.4533329 0.04827632 0.6955776 0.04080962 0.51514808 0.22959696 0.01200912
## 100 0.8288251 0.42112852 0.7027307 0.11225748 0.32787879 0.72925445 0.77385391
##             V8         V9       V10         V11
## 1   0.44567883 0.05624119 0.1115611 0.007690465
## 2   0.51813536 0.80437551 0.7154438 0.695098008
```

```
## 3   0.60643171 0.16521569 0.3467028 0.886558322
## 4   0.03410814 0.89823535 0.1709313 0.357205148
## 5   0.02692736 0.41044409 0.1550050 0.893121126
## 6   0.80070102 0.20813039 0.3052561 0.305675219
## 7   0.63822954 0.62213635 0.1083129 0.631312733
## 8   0.55184616 0.92621359 0.6604291 0.473794648
## 9   0.97658445 0.69807583 0.2291064 0.874133293
## 10  0.53916084 0.48083269 0.2588800 0.245211026
## 11  0.44567883 0.05624119 0.1115611 0.007690465
## 12  0.51813536 0.80437551 0.7154438 0.695098008
## 13  0.60643171 0.16521569 0.3467028 0.886558322
## 14  0.03410814 0.89823535 0.1709313 0.357205148
## 15  0.02692736 0.41044409 0.1550050 0.893121126
## 16  0.80070102 0.20813039 0.3052561 0.305675219
## 17  0.63822954 0.62213635 0.1083129 0.631312733
## 18  0.55184616 0.92621359 0.6604291 0.473794648
## 19  0.97658445 0.69807583 0.2291064 0.874133293
## 20  0.53916084 0.48083269 0.2588800 0.245211026
## 21  0.44567883 0.05624119 0.1115611 0.007690465
## 22  0.51813536 0.80437551 0.7154438 0.695098008
## 23  0.60643171 0.16521569 0.3467028 0.886558322
## 24  0.03410814 0.89823535 0.1709313 0.357205148
## 25  0.02692736 0.41044409 0.1550050 0.893121126
## 26  0.80070102 0.20813039 0.3052561 0.305675219
## 27  0.63822954 0.62213635 0.1083129 0.631312733
## 28  0.55184616 0.92621359 0.6604291 0.473794648
## 29  0.97658445 0.69807583 0.2291064 0.874133293
## 30  0.53916084 0.48083269 0.2588800 0.245211026
## 31  0.44567883 0.05624119 0.1115611 0.007690465
## 32  0.51813536 0.80437551 0.7154438 0.695098008
## 33  0.60643171 0.16521569 0.3467028 0.886558322
## 34  0.03410814 0.89823535 0.1709313 0.357205148
## 35  0.02692736 0.41044409 0.1550050 0.893121126
## 36  0.80070102 0.20813039 0.3052561 0.305675219
## 37  0.63822954 0.62213635 0.1083129 0.631312733
## 38  0.55184616 0.92621359 0.6604291 0.473794648
## 39  0.97658445 0.69807583 0.2291064 0.874133293
## 40  0.53916084 0.48083269 0.2588800 0.245211026
## 41  0.44567883 0.05624119 0.1115611 0.007690465
## 42  0.51813536 0.80437551 0.7154438 0.695098008
## 43  0.60643171 0.16521569 0.3467028 0.886558322
## 44  0.03410814 0.89823535 0.1709313 0.357205148
## 45  0.02692736 0.41044409 0.1550050 0.893121126
## 46  0.80070102 0.20813039 0.3052561 0.305675219
## 47  0.63822954 0.62213635 0.1083129 0.631312733
## 48  0.55184616 0.92621359 0.6604291 0.473794648
## 49  0.97658445 0.69807583 0.2291064 0.874133293
## 50  0.53916084 0.48083269 0.2588800 0.245211026
## 51  0.44567883 0.05624119 0.1115611 0.007690465
## 52  0.51813536 0.80437551 0.7154438 0.695098008
## 53  0.60643171 0.16521569 0.3467028 0.886558322
## 54  0.03410814 0.89823535 0.1709313 0.357205148
## 55  0.02692736 0.41044409 0.1550050 0.893121126
## 56  0.80070102 0.20813039 0.3052561 0.305675219
```

```
## 57  0.63822954 0.62213635 0.1083129 0.631312733
## 58  0.55184616 0.92621359 0.6604291 0.473794648
## 59  0.97658445 0.69807583 0.2291064 0.874133293
## 60  0.53916084 0.48083269 0.2588800 0.245211026
## 61  0.44567883 0.05624119 0.1115611 0.007690465
## 62  0.51813536 0.80437551 0.7154438 0.695098008
## 63  0.60643171 0.16521569 0.3467028 0.886558322
## 64  0.03410814 0.89823535 0.1709313 0.357205148
## 65  0.02692736 0.41044409 0.1550050 0.893121126
## 66  0.80070102 0.20813039 0.3052561 0.305675219
## 67  0.63822954 0.62213635 0.1083129 0.631312733
## 68  0.55184616 0.92621359 0.6604291 0.473794648
## 69  0.97658445 0.69807583 0.2291064 0.874133293
## 70  0.53916084 0.48083269 0.2588800 0.245211026
## 71  0.44567883 0.05624119 0.1115611 0.007690465
## 72  0.51813536 0.80437551 0.7154438 0.695098008
## 73  0.60643171 0.16521569 0.3467028 0.886558322
## 74  0.03410814 0.89823535 0.1709313 0.357205148
## 75  0.02692736 0.41044409 0.1550050 0.893121126
## 76  0.80070102 0.20813039 0.3052561 0.305675219
## 77  0.63822954 0.62213635 0.1083129 0.631312733
## 78  0.55184616 0.92621359 0.6604291 0.473794648
## 79  0.97658445 0.69807583 0.2291064 0.874133293
## 80  0.53916084 0.48083269 0.2588800 0.245211026
## 81  0.44567883 0.05624119 0.1115611 0.007690465
## 82  0.51813536 0.80437551 0.7154438 0.695098008
## 83  0.60643171 0.16521569 0.3467028 0.886558322
## 84  0.03410814 0.89823535 0.1709313 0.357205148
## 85  0.02692736 0.41044409 0.1550050 0.893121126
## 86  0.80070102 0.20813039 0.3052561 0.305675219
## 87  0.63822954 0.62213635 0.1083129 0.631312733
## 88  0.55184616 0.92621359 0.6604291 0.473794648
## 89  0.97658445 0.69807583 0.2291064 0.874133293
## 90  0.53916084 0.48083269 0.2588800 0.245211026
## 91  0.44567883 0.05624119 0.1115611 0.007690465
## 92  0.51813536 0.80437551 0.7154438 0.695098008
## 93  0.60643171 0.16521569 0.3467028 0.886558322
## 94  0.03410814 0.89823535 0.1709313 0.357205148
## 95  0.02692736 0.41044409 0.1550050 0.893121126
## 96  0.80070102 0.20813039 0.3052561 0.305675219
## 97  0.63822954 0.62213635 0.1083129 0.631312733
## 98  0.55184616 0.92621359 0.6604291 0.473794648
## 99  0.97658445 0.69807583 0.2291064 0.874133293
## 100 0.53916084 0.48083269 0.2588800 0.245211026
```

```r
write.csv(x = df_doe1,file="df_doe1.csv", row.names = F)
```

Analysis

Let's get the corresponding results now (session has changed now...):

```
df=read.csv("Data",header=T)
str(df)
```

```
## 'data.frame':    66 obs. of  13 variables:
##  $ Date: chr  "2023-12-24-22:09:55" "2024-01-11-01:54:36" "2024-01-11-01:55:42" "2024-01-11-01:55:43
##  $ x1  : num  0.1 0.9 0.9 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x2  : num  0.2 0.4 0.4 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x3  : num  0.2 0.99 0.99 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x4  : num  0.3 0.9 0.9 0.99 0 0.5 0.5 0.5 0.5 0.5 ...
##  $ x5  : num  0.5 0.88 0.88 0.99 0.9 0.4 0.4 0.4 0.4 0.4 ...
##  $ x6  : num  0.6 0.44 0.44 0.99 0.9 0.5 0.5 0.5 0.5 0.5 ...
##  $ x7  : num  0.7 0.55 0.55 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x8  : num  0.01 0.7 0.7 0.99 0.9 0.8 0.8 0.8 0.8 0.8 ...
##  $ x9  : num  0.4 0.8 0.8 0.99 0.9 0.3 0.3 0.3 0.3 0.3 ...
##  $ x10 : num  0.44 0.7 0.7 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x11 : num  0.6 0.9 0.9 0.99 0.9 0.6 0.6 0.6 0.6 0.6 ...
##  $ y   : num  0.477 0.359 0.356 -0.828 -0.599 ...
```

```
df = df[41:60,]
```

Now let's evaluate mean and variability for each combination

```
df %>% mutate(label=paste0(x1,"_",x2,"_",x3,"_",x4,"_",x5,"_",x6,"_",x7,"_",x8,"_",x9,"_",x10,"_",x11))
```

```
## # A tibble: 17 x 3
##    label                              mean       sd
##    <chr>                             <dbl>    <dbl>
##  1 0.1_0_0_0_0_0_0_0_0_0_0           1.04  NA
##  2 0.6_0_0_0.6_0_0.6_0_0.6_0_0.1_0.6 2.74  NA
##  3 0.6_0_0_0.7_0_0.6_0_0.6_0_0.1_0.6 2.84  NA
##  4 0.6_0_0_0_0_0_0_0_0_0_0           2.15  NA
##  5 0.7_0_0.7_0_0.7_0_0.7_0_0.7_0_0.7 1.26  NA
##  6 0.7_0_0_0.1_0_0.1_0_0.1_0_0.1_0.7 2.75  NA
##  7 0.7_0_0_0.7_0_0.7_0_0.7_0_0.1_0.6 3.35  NA
##  8 0.7_0_0_0_0_0_0_0_0_0.1_0.7       2.65  NA
##  9 0.7_0_0_0_0_0_0_0_0_0_0           2.65   0.00356
## 10 0.7_0_0_0_0_0_0_0_0_0_0.7         2.66  NA
## 11 0.8_0_0_0.8_0_0.8_0_0.88_0_0.1_0  3.25  NA
## 12 0.8_0_0_0.8_0_0.8_0_0.88_0_0.1_0.1 3.25  NA
## 13 0.8_0_0_0.8_0_0.8_0_0.88_0_0.1_0.8 3.25  NA
## 14 0.8_0_0_0_0_0_0_0_0_0.1_0.8       2.45  NA
## 15 0_0_0_0_0_0_0_0_0_0              1.02  NA
## 16 0_0_0_0_0_0_0_0_0_0_1            1.01  NA
## 17 1_0_0_0_0_0_0_0_0_0_0            0.419 NA
```

All right, variability depending on x appears quite large (and coherent with previous observations) compared to "measurement" variability (i.e., in [-1:3] compared to [0, 6e-03].) Furthermore variability does not appear to depend too much on inputs so pseudo-replication (measuring the same combination of values) should be of little use compared to replication (really randomizing inputs). The hardest part will thus be to come up with a decent model for f.

Influence of parameters

DoE

Working in dimension 11 is way to large, let's try to evaluate which parameters are influent with a screening design. `FrF2` is the right package for this but classical fractionnal designs are probably too conservative so we'll play with `pb` instead. I had to install `FrF2` through `install.packages` as it is not in debian. :(

```
library(FrF2)
```

```
## Warning: package 'FrF2' was built under R version 4.3.2
```

```
## Loading required package: DoE.base
```

```
## Warning: package 'DoE.base' was built under R version 4.3.2
```

```
## Loading required package: grid
```

```
## Loading required package: conf.design
```

```
## Registered S3 method overwritten by 'DoE.base':
##   method           from
##   factorize.factor conf.design
```

```
##
## Attaching package: 'DoE.base'
```

```
## The following objects are masked from 'package:stats':
##
##     aov, lm
```

```
## The following object is masked from 'package:graphics':
##
##     plot.design
```

```
## The following object is masked from 'package:base':
##
##     lengths
```

Let's call `pb` several times in a row to concatenate several balanced designs.

```
space_dim = 11
n_repeat = 5
n_replicates = 1
df_doe2 = data.frame()

for(i in 1:n_repeat) {
  df_doe2 = rbind(df_doe2,
                  pb(nruns = space_dim+1, default.levels = c(0,1), factor.names = paste0("x",1:11),
                     replications = n_replicates, seed = 415234+i))
}
df_doe2
```

```
##    x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11
## 1   0  1  0  1  1  0  1  1  1   0   0
## 2   0  0  0  1  0  1  1  0  1   1   1
## 3   0  1  1  0  1  1  1  0  0   0   1
## 4   1  1  0  0  0  1  0  1  1   0   1
## 5   1  0  1  1  0  1  1  1  0   0   0
## 6   1  0  0  0  1  0  1  1  0   1   1
## 7   0  0  0  0  0  0  0  0  0   0   0
## 8   0  1  1  1  0  0  0  1  0   1   1
## 9   1  1  0  1  1  1  0  0  0   1   0
## 10  0  0  1  0  1  1  0  1  1   1   0
## 11  1  1  1  0  0  0  1  0  1   1   0
## 12  1  0  1  1  1  0  0  0  1   0   1
## 13  1  0  1  1  1  0  0  0  1   0   1
## 14  0  1  0  1  1  0  1  1  1   0   0
## 15  1  1  0  0  0  1  0  1  1   0   1
## 16  0  0  0  0  0  0  0  0  0   0   0
## 17  1  1  1  0  0  0  1  0  1   1   0
## 18  1  0  0  0  1  0  1  1  0   1   1
## 19  0  0  1  0  1  1  0  1  1   1   0
## 20  1  0  1  1  0  1  1  1  0   0   0
## 21  0  1  1  1  0  0  0  1  0   1   1
## 22  1  1  0  1  1  1  0  0  0   1   0
## 23  0  0  0  1  0  1  1  0  1   1   1
## 24  0  1  1  0  1  1  1  0  0   0   1
## 25  1  1  0  0  0  1  0  1  1   0   1
## 26  0  1  1  0  1  1  1  0  0   0   1
## 27  0  0  0  1  0  1  1  0  1   1   1
## 28  0  1  1  1  0  0  0  1  0   1   1
## 29  0  0  0  0  0  0  0  0  0   0   0
## 30  1  1  0  1  1  1  0  0  0   1   0
## 31  1  0  1  1  1  0  0  0  1   0   1
## 32  1  1  1  0  0  0  1  0  1   1   0
## 33  0  0  1  0  1  1  0  1  1   1   0
## 34  1  0  1  1  0  1  1  1  0   0   0
## 35  1  0  0  0  1  0  1  1  0   1   1
## 36  0  1  0  1  1  0  1  1  1   0   0
## 37  1  1  0  1  1  1  0  0  0   1   0
## 38  1  0  1  1  1  0  0  0  1   0   1
## 39  1  0  0  0  1  0  1  1  0   1   1
## 40  0  0  0  1  0  1  1  0  1   1   1
## 41  0  0  1  0  1  1  0  1  1   1   0
## 42  1  1  1  0  0  0  1  0  1   1   0
## 43  1  1  0  0  0  1  0  1  1   0   1
## 44  0  1  1  0  1  1  1  0  0   0   1
## 45  0  1  1  1  0  0  0  1  0   1   1
## 46  0  0  0  0  0  0  0  0  0   0   0
## 47  0  1  0  1  1  0  1  1  1   0   0
## 48  1  0  1  1  0  1  1  1  0   0   0
## 49  0  0  1  0  1  1  0  1  1   1   0
## 50  0  0  0  1  0  1  1  0  1   1   1
## 51  1  1  1  0  0  0  1  0  1   1   0
## 52  1  1  0  0  0  1  0  1  1   0   1
## 53  1  0  1  1  1  0  0  0  1   0   1
```

```
## 54  1  1  0  1  1  1  0  0  0   1   0
## 55  1  0  1  1  0  1  1  1  0   0   0
## 56  0  1  1  1  0  0  0  1  0   1   1
## 57  0  0  0  0  0  0  0  0  0   0   0
## 58  0  1  1  0  1  1  1  0  0   0   1
## 59  1  0  0  0  1  0  1  1  0   1   1
## 60  0  1  0  1  1  0  1  1  1   0   0
## class=design, type= pb
```

Let's check how many combinations I actually generated.

```
df_doe2 %>% mutate(label=paste0(x1,"_",x2,"_",x3,"_",x4,"_",x5,"_",x6,"_",x7,"_",x8,"_",x9,"_",x10,"_",x
```

```
##                    label
## 1  0_1_0_1_1_0_1_1_1_0_0
## 2  0_0_0_1_0_1_1_0_1_1_1
## 3  0_1_1_0_1_1_1_0_0_0_1
## 4  1_1_0_0_0_1_0_1_1_0_1
## 5  1_0_1_1_0_1_1_1_0_0_0
## 6  1_0_0_0_1_0_1_1_0_1_1
## 7  0_0_0_0_0_0_0_0_0_0_0
## 8  0_1_1_1_0_0_0_1_0_1_1
## 9  1_1_0_1_1_1_0_0_0_1_0
## 10 0_0_1_0_1_1_0_1_1_1_0
## 11 1_1_1_0_0_0_1_0_1_1_0
## 12 1_0_1_1_1_0_0_0_1_0_1
```

This fails, randomization in `pb` is only for the order, not for the values. Let's use dummy variables instead.

```
space_dim = 11
n_repeat = 5
n_replicates = 1
df_doe2 = pb(nruns = n_repeat*(space_dim+1), default.levels = c(0,1), factor.names = paste0("x",1:11),
                    replications = n_replicates, seed = 415234)
df_doe2
```

```
##    x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11 e1 e2 e3 e4 e5 e6 e7 e8 e9 e10 e11 e12
## 1   0  1  1  1  1  0  0  1  1   1   1  1  0  0  0  0  0  1  1  0   0   0   0
## 2   0  0  1  1  1  1  1  0  0   0   0  0  1  1  0  0  0  0  1  0   0   0   1
## 3   0  1  0  0  0  1  0  1  1   0   1  1  1  0  1  0  1  0  0  1   0   0   1
## 4   1  1  1  1  1  0  0  0  0   0   1  1  0  0  0  0  1  0  0  0   1   1   0
## 5   0  0  1  1  0  0  0  0  1   0   0  0  1  1  0  1  1  0  1  0   1   0   0
## 6   0  0  0  1  1  0  1  1  0   1   0  1  0  0  0  1  0  1  1  0   1   1   1
## 7   0  1  0  0  1  0  0  1  1   1   0  1  1  1  1  0  0  1  1  1   1   1   0
## 8   0  1  1  0  1  1  1  0  1   0   1  0  0  1  0  0  1  1  1  0   1   1   1
## 9   1  1  1  0  1  0  1  0  0   1   0  0  1  1  1  0  1  1  1  1   0   0   1
## 10  0  1  0  1  1  0  1  1  1   0   1  0  1  0  0  1  0  0  1  1   1   0   1
## 11  1  1  0  1  1  0  1  0  1   0   0  0  1  0  1  1  0  1  1  1   0   1   0
## 12  0  0  1  0  0  0  1  1  0   1   1  0  1  0  1  0  0  0  1  0   1   1   0
## 13  0  0  0  0  0  0  0  0  0   0   0  0  0  0  0  0  0  0  0  0   0   0   0
## 14  1  0  1  1  1  0  1  0  1   0   0  1  0  0  1  1  1  0  1  1   1   1   0
## 15  0  1  1  1  1  1  0  0  0   0   0  1  1  0  0  0  0  1  0  0   0   1   1
```

```
## 16  1  0  0  1  1  1  0  1  1   1   1  0  0  1  1  1  1  1  0  0   0   0   0
## 17  1  0  0  1  1  1  1  1  0   0   0  0  0  1  1  0  0  0  0  1   0   0   0
## 18  1  1  0  1  1  1  0  1  0   1   0  0  1  0  0  1  1  1  0  1   1   1   1
## 19  1  1  1  1  0  0  1  1  1   1   1  0  0  0  0  0  1  1  0  0   0   0   1
## 20  1  1  1  0  1  1  1  1  0   0   1  1  1  1  1  0  0  0  0  0   1   1   0
## 21  1  1  0  1  0  1  0  0  0   1   0  1  1  0  1  1  1  0  1  0   1   0   0
## 22  0  0  1  1  1  0  1  1  1   1   0  0  1  1  1  1  1  0  0  0   0   0   1
## 23  1  0  1  1  0  1  0  1  0   0   0  1  0  1  1  0  1  1  1  0   1   0   1
## 24  1  0  1  0  1  0  0  1  0   0   1  1  1  0  1  1  1  1  0  0   1   1   1
## 25  1  1  0  1  0  1  0  0  1   0   0  1  1  1  0  1  1  1  1  0   0   1   1
## 26  1  0  0  1  0  0  1  1  1   0   1  1  1  1  0  0  1  1  1  1   1   0   0
## 27  0  1  0  0  0  1  1  0  1   1   0  1  0  1  0  0  0  1  0  1   1   0   1
## 28  0  0  0  1  1  0  0  0  0   1   0  0  0  1  1  0  1  1  0  1   0   1   0
## 29  1  1  0  0  1  1  1  1  1   0   0  0  0  0  1  1  0  0  0  0   1   0   0
## 30  0  1  1  0  0  0  0  1  0   0   0  1  1  0  1  1  0  1  0  1   0   0   0
## 31  1  0  1  0  0  0  1  0  1   1   0  1  1  1  0  1  0  1  0  0   1   0   0
## 32  0  0  0  0  1  0  0  0  1   1   0  1  1  0  1  0  1  0  0  0   1   0   1
## 33  1  1  1  0  0  0  0  0  1   1   0  0  0  0  1  0  0  0  1  1   0   1   1
## 34  1  1  0  1  1  1  1  0  0   1   1  1  1  1  0  0  0  0  0  1   1   0   0
## 35  1  0  1  0  1  0  0  0  1   0   1  1  0  1  1  1  0  1  0  1   0   0   1
## 36  0  0  1  0  0  1  1  1  0   1   1  1  1  0  0  1  1  1  1  1   0   0   0
## 37  1  1  0  0  0  0  1  0  0   0   1  1  0  1  1  0  1  0  1  0   0   0   1
## 38  1  1  1  1  0  0  0  0  0   1   1  0  0  0  0  1  0  0  0  1   1   0   1
## 39  0  1  1  1  0  1  1  1  1   0   0  1  1  1  1  1  0  0  0  0   0   1   1
## 40  0  0  1  1  0  1  1  0  1   0   1  0  0  0  1  0  1  1  0  1   1   1   0
## 41  0  1  0  1  0  0  1  0  0   1   1  1  0  1  1  1  1  0  0  1   1   1   1
## 42  0  1  1  0  1  1  0  1  0   1   0  0  0  1  0  1  1  0  1  1   1   0   1
## 43  1  0  1  1  1  1  0  0  1   1   1  1  1  0  0  0  0  0  1  1   0   0   0
## 44  0  1  0  0  1  1  1  0  1   1   1  1  0  0  1  1  1  1  1  0   0   0   0
## 45  0  1  0  1  0  0  0  1  0   1   1  0  1  1  1  0  1  0  1  0   0   1   0
## 46  1  1  1  0  0  1  1  1  1   1   0  0  0  0  0  1  1  0  0  0   0   1   0
## 47  0  0  0  0  0  1  1  0  0   0   0  1  0  0  0  1  1  0  1  1   0   1   0
## 48  1  0  0  0  1  1  0  1  1   0   1  0  1  0  0  0  1  0  1  1   0   1   1
## 49  0  1  1  0  1  0  1  0  0   0   1  0  1  1  0  1  1  1  0  1   0   1   0
## 50  0  0  0  1  0  1  1  0  1   1   1  0  1  0  1  0  0  1  0  0   1   1   1
## 51  0  1  1  1  0  1  0  1  0   0   1  0  0  1  1  1  0  1  1  1   1   0   0
## 52  1  0  1  1  0  1  1  1  0   1   0  1  0  0  1  0  0  1  1  1   0   1   1
## 53  1  0  0  0  0  1  0  0  0   1   1  0  1  1  0  1  0  1  0  0   0   1   0
## 54  1  1  0  0  0  0  0  1  1   0   0  0  0  1  0  0  0  1  1  0   1   1   0
## 55  0  0  0  1  0  0  0  1  1   0   1  1  0  1  0  1  0  0  0  1   0   1   1
## 56  1  0  1  0  0  1  0  0  1   1   1  0  1  1  1  1  0  0  1  1   1   1   1
## 57  1  0  0  0  1  0  1  1  0   1   1  1  0  1  0  1  0  0  1  0   0   1   1
## 58  0  0  1  0  1  1  0  1  1   1   0  1  0  1  0  0  1  0  0  1   1   1   0
## 59  1  0  0  0  0  0  1  1  0   0   0  0  1  0  0  0  1  1  0  1   1   0   1
## 60  0  0  0  0  1  1  0  0  0   0   1  0  0  0  1  1  0  1  1  0   1   0   1
##    e13 e14 e15 e16 e17 e18 e19 e20 e21 e22 e23 e24 e25 e26 e27 e28 e29 e30 e31
## 1    1   0   0   0   1   1   0   1   1   0   1   0   1   0   0   0   1   0   1
## 2    1   0   1   1   0   1   0   1   0   0   0   1   0   1   1   0   1   1   1
## 3    1   1   0   1   1   1   1   0   0   1   1   1   1   1   0   0   0   0   0
## 4    1   1   0   1   0   1   0   0   0   1   0   1   1   0   1   1   1   0   1
## 5    0   1   0   1   1   0   1   1   1   0   1   0   1   0   0   1   0   0   1
## 6    0   1   0   1   0   0   1   0   0   1   1   1   0   1   1   1   1   0   0
## 7    0   0   0   0   1   1   0   0   0   0   1   0   0   0   1   1   0   1   1
## 8    1   0   0   1   1   1   1   1   0   0   0   0   0   1   1   0   0   0   0
```

```
## 9    1   1   1   1   0   0   0   0   0   1   1   0   0   0   0   1   0   0   0
## 10   1   1   1   0   0   1   1   1   1   1   0   0   0   0   0   1   1   0   0
## 11   1   0   0   1   0   0   1   1   1   0   1   1   1   1   0   0   1   1   1
## 12   1   1   1   0   1   0   1   0   0   1   0   0   1   1   1   0   1   1   1
## 13   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 14   0   1   1   1   1   1   0   0   0   0   0   1   1   0   0   0   0   1   0
## 15   0   1   1   0   1   0   1   0   0   0   1   0   1   1   0   1   1   1   0
## 16   1   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0   1   0   1
## 17   1   1   0   1   1   0   1   0   1   0   0   0   1   0   1   1   0   1   1
## 18   0   0   1   1   1   1   1   0   0   0   0   0   1   1   0   0   0   0   1
## 19   0   0   0   1   1   0   1   1   0   1   0   1   0   0   0   1   0   1   1
## 20   0   0   0   1   0   0   0   1   1   0   1   1   0   1   0   1   0   0   0
## 21   1   0   0   1   1   1   0   1   1   1   1   0   0   1   1   1   1   1   0
## 22   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0   1   0   1   0
## 23   0   0   1   0   0   1   1   1   0   1   1   1   1   0   0   1   1   1   1
## 24   1   1   0   0   0   0   0   1   1   0   0   0   0   1   0   0   0   1   1
## 25   1   1   1   0   0   0   0   0   1   1   0   0   0   0   1   0   0   0   1
## 26   0   0   0   1   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0
## 27   1   1   0   1   0   1   0   0   1   0   0   1   1   1   0   1   1   1   1
## 28   0   0   1   0   1   1   0   1   1   1   0   1   0   1   0   0   1   0   0
## 29   0   1   1   0   1   1   0   1   0   1   0   0   0   1   0   1   1   0   1
## 30   1   0   1   1   0   1   1   1   0   1   0   1   0   0   1   0   0   1   1
## 31   1   1   1   0   1   1   1   1   0   0   1   1   1   1   1   0   0   0   0
## 32   1   0   1   1   1   0   1   0   1   0   0   1   0   0   1   1   1   0   1
## 33   0   1   0   1   0   0   0   1   0   1   1   0   1   1   1   0   1   0   1
## 34   0   0   1   0   0   0   1   1   0   1   1   0   1   0   1   0   0   0   1
## 35   0   0   1   1   1   0   1   1   1   1   0   0   1   1   1   1   1   0   0
## 36   0   0   1   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0   1
## 37   0   1   1   0   1   1   1   0   1   0   1   0   0   1   0   0   1   1   1
## 38   1   0   1   0   1   0   0   0   1   0   1   1   0   1   1   1   0   1   0
## 39   0   0   0   0   1   0   0   0   1   1   0   1   1   0   1   0   1   0   0
## 40   1   0   1   0   0   1   0   0   1   1   1   0   1   1   1   1   0   0   1
## 41   1   0   0   0   0   0   1   1   0   0   0   0   1   0   0   0   1   1   0
## 42   0   1   0   0   1   0   0   1   1   1   0   1   1   1   1   0   0   1   1
## 43   0   1   0   0   0   1   1   0   1   1   0   1   0   1   0   0   0   1   0
## 44   0   1   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0   1   0
## 45   0   1   1   1   0   1   1   1   1   0   0   1   1   1   1   1   0   0   0
## 46   0   0   1   1   0   1   1   0   1   0   1   0   0   0   1   0   1   1   0
## 47   1   0   0   0   1   0   1   1   0   1   1   1   0   1   0   1   0   0   1
## 48   1   0   1   0   1   0   0   1   0   0   1   1   1   0   1   1   1   1   0
## 49   0   1   0   0   1   1   1   0   1   1   1   1   0   0   1   1   1   1   1
## 50   0   1   1   1   1   0   0   1   1   1   1   1   0   0   0   0   0   1   1
## 51   1   1   1   1   1   0   0   0   0   0   1   1   0   0   0   0   1   0   0
## 52   1   1   0   0   1   1   1   1   1   0   0   0   0   0   1   1   0   0   0
## 53   1   1   0   1   1   1   0   1   0   1   0   0   1   0   0   1   1   1   0
## 54   1   0   1   0   0   0   1   0   1   1   0   1   1   1   0   1   0   1   0
## 55   0   1   1   1   0   1   0   1   0   0   1   0   0   1   1   1   0   1   1
## 56   0   0   0   0   0   1   1   0   0   0   0   1   0   0   0   1   1   0   1
## 57   1   0   1   1   1   1   0   0   1   1   1   1   1   0   0   0   0   0   1
## 58   1   1   1   1   0   0   1   1   1   1   1   0   0   0   0   0   1   1   0
## 59   0   1   0   0   0   1   0   1   1   0   1   1   1   0   1   0   1   0   0
## 60   0   0   0   1   0   1   1   0   1   1   1   0   1   0   1   0   0   1   0
##     e32 e33 e34 e35 e36 e37 e38 e39 e40 e41 e42 e43 e44 e45 e46 e47 e48
## 1    1   0   1   1   1   0   1   0   1   0   0   1   0   0   1   1   1
```

```
## 2     0   1   0   1   0   0   1   0   0   1   1   1   0   1   1   1   1
## 3     1   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0   1
## 4     0   1   0   0   1   0   0   1   1   1   0   1   1   1   1   0   0
## 5     1   1   0   1   1   1   1   0   0   1   1   1   1   1   0   0   0
## 6     1   1   1   1   1   0   0   0   0   0   1   1   0   0   0   0   1
## 7     0   1   0   1   0   0   0   1   0   1   1   0   1   1   1   0   1
## 8     1   0   0   0   1   1   0   1   1   0   1   0   1   0   0   0   1
## 9     1   1   0   1   1   0   1   0   1   0   0   0   1   0   1   1   0
## 10    0   0   1   0   0   0   1   1   0   1   1   0   1   0   1   0   0
## 11    1   1   0   0   0   0   0   1   1   0   0   0   0   1   0   0   0
## 12    1   0   0   1   1   1   1   1   0   0   0   0   0   1   1   0   0
## 13    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 14    0   0   1   1   0   1   1   0   1   0   1   0   0   0   1   0   1
## 15    1   0   1   0   0   1   0   0   1   1   1   0   1   1   1   1   0
## 16    0   0   0   1   0   1   1   0   1   1   1   0   1   0   1   0   0
## 17    1   0   1   0   1   0   0   1   0   0   1   1   1   0   1   1   1
## 18    0   0   0   1   1   0   1   1   0   1   0   1   0   0   0   1   0
## 19    0   1   1   1   0   1   0   1   0   0   1   0   0   1   1   1   0
## 20    1   0   1   1   0   1   1   1   0   1   0   1   0   0   1   0   0
## 21    0   0   0   0   1   1   0   0   0   0   1   0   0   0   1   1   0
## 22    0   0   1   0   1   1   0   1   1   1   0   1   0   1   0   0   1
## 23    1   0   0   0   0   0   1   1   0   0   0   0   1   0   0   0   1
## 24    0   1   1   0   1   0   1   0   0   0   1   0   1   1   0   1   1
## 25    1   0   1   1   0   1   0   1   0   0   0   1   0   1   1   0   1
## 26    1   0   1   0   0   0   1   0   1   1   0   1   1   1   0   1   0
## 27    0   0   1   1   1   1   1   0   0   0   0   0   1   1   0   0   0
## 28    1   1   1   0   1   1   1   1   0   0   1   1   1   1   1   0   0
## 29    1   1   0   1   0   1   0   0   1   0   0   1   1   1   0   1   1
## 30    1   0   1   1   1   1   0   0   1   1   1   1   1   0   0   0   0
## 31    0   1   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0
## 32    1   1   1   0   0   1   1   1   1   1   0   0   0   0   0   1   1
## 33    0   0   1   0   0   1   1   1   0   1   1   1   1   0   0   1   1
## 34    0   1   1   0   1   1   1   0   1   0   1   0   0   1   0   0   1
## 35    0   0   0   1   1   0   0   0   0   1   0   0   0   1   1   0   1
## 36    0   1   0   0   0   1   0   1   1   0   1   1   1   0   1   0   1
## 37    0   1   1   1   1   0   0   1   1   1   1   1   0   0   0   0   0
## 38    1   0   0   1   0   0   1   1   1   0   1   1   1   1   0   0   1
## 39    0   1   0   1   1   0   1   1   1   0   1   0   1   0   0   1   0
## 40    1   1   1   1   0   0   0   0   0   1   1   0   0   0   0   1   0
## 41    1   1   0   1   0   1   0   0   0   1   0   1   1   0   1   1   1
## 42    1   1   1   0   0   0   0   0   1   1   0   0   0   0   1   0   0
## 43    1   1   0   1   1   1   0   1   0   1   0   0   1   0   0   1   1
## 44    1   0   0   0   1   0   1   1   0   1   1   1   0   1   0   1   0
## 45    0   0   1   1   0   0   0   0   1   0   0   0   1   1   0   1   1
## 46    1   1   1   0   1   0   1   0   0   1   0   0   1   1   1   0   1
## 47    0   0   1   1   1   0   1   1   1   1   0   0   1   1   1   1   1
## 48    0   1   1   1   1   1   0   0   0   0   0   1   1   0   0   0   0
## 49    0   0   0   0   0   1   1   0   0   0   0   1   0   0   0   1   1
## 50    0   0   0   0   1   0   0   0   1   1   0   1   1   0   1   0   1
## 51    0   1   1   0   1   1   0   1   0   1   0   0   0   1   0   1   1
## 52    0   1   0   0   0   1   1   0   1   1   0   1   0   1   0   0   0
## 53    1   1   1   1   0   0   1   1   1   1   1   0   0   0   0   0   1
## 54    0   1   0   0   1   1   1   0   1   1   1   1   0   0   1   1   1
## 55    1   1   0   0   1   1   1   1   1   0   0   0   0   0   1   1   0
```

```
## 56   1   0   1   0   1   0   0   0   1   0   1   1   0   1   1   1   0
## 57   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0   1   0
## 58   0   0   0   1   0   0   0   1   1   0   1   1   0   1   0   1   0
## 59   1   0   0   1   1   1   0   1   1   1   1   0   0   1   1   1   1
## 60   0   1   1   1   0   1   1   1   1   0   0   1   1   1   1   1   0
## class=design, type= pb
```

```
d = df_doe2 %>% mutate(label=paste0(x1,"_",x2,"_",x3,"_",x4,"_",x5,"_",x6,"_",x7,"_",x8,"_",x9,"_",x10,
unique(d$label)
```

```
##  [1] "0_1_1_1_1_0_0_1_1_1_1" "0_0_1_1_1_1_1_0_0_0_0" "0_1_0_0_0_1_0_1_1_0_1"
##  [4] "1_1_1_1_1_0_0_0_0_0_1" "0_0_1_1_0_0_0_0_1_0_0" "0_0_0_1_1_0_1_1_0_1_0"
##  [7] "0_1_0_0_1_0_0_1_1_1_0" "0_1_1_0_1_1_1_0_1_0_1" "1_1_1_0_1_0_1_0_0_1_0"
## [10] "0_1_0_1_1_0_1_1_1_0_1" "1_1_0_1_1_0_1_0_1_0_0" "0_0_1_0_0_0_1_1_0_1_1"
## [13] "0_0_0_0_0_0_0_0_0_0_0" "1_0_1_1_1_0_1_0_1_0_0" "0_1_1_1_1_1_0_0_0_0_0"
## [16] "1_0_0_1_1_1_0_1_1_1_1" "1_0_0_1_1_1_1_1_0_0_0" "1_1_0_1_1_1_0_1_0_1_0"
## [19] "1_1_1_1_0_0_1_1_1_1_1" "1_1_1_0_1_1_1_1_0_0_1" "1_1_0_1_0_1_0_0_0_1_0"
## [22] "0_0_1_1_1_0_1_1_1_1_0" "1_0_1_1_0_1_0_1_0_0_0" "1_0_1_0_1_0_0_1_0_0_1"
## [25] "1_1_0_1_0_1_0_0_1_0_0" "1_0_0_1_0_0_1_1_1_0_1" "0_1_0_0_0_1_1_0_1_1_0"
## [28] "0_0_0_1_1_0_0_0_0_1_0" "1_1_0_0_1_1_1_1_1_0_0" "0_1_1_0_0_0_0_1_0_0_0"
## [31] "1_0_1_0_0_0_1_0_1_1_0" "0_0_0_0_1_0_0_0_1_1_0" "1_1_1_0_0_0_0_0_0_1_0"
## [34] "1_1_0_1_1_1_1_0_0_1_1" "1_0_1_0_1_0_0_0_1_0_1" "0_0_1_0_0_1_1_1_0_1_1"
## [37] "1_1_0_0_0_0_1_0_0_0_1" "1_1_1_1_0_0_0_0_0_1_1" "0_1_1_1_0_1_1_1_1_0_0"
## [40] "0_0_1_1_0_1_1_0_1_0_1" "0_1_0_1_0_0_1_0_0_1_1" "0_1_1_0_1_1_0_1_0_1_0"
## [43] "1_0_1_1_1_1_0_0_1_1_1" "0_1_0_0_1_1_1_0_1_1_1" "0_1_0_1_0_0_0_1_0_1_1"
## [46] "1_1_1_0_0_1_1_1_1_1_0" "0_0_0_0_0_1_1_0_0_0_0" "1_0_0_0_1_1_0_1_1_0_1"
## [49] "0_1_1_0_1_0_1_0_0_0_1" "0_0_0_1_0_1_1_0_1_1_1" "0_1_1_1_0_1_0_1_0_0_1"
## [52] "1_0_1_1_0_1_1_1_0_1_0" "1_0_0_0_0_1_0_0_0_1_1" "1_1_0_0_0_0_0_0_1_1_0"
## [55] "0_0_0_1_0_0_0_1_1_0_1" "1_0_1_0_0_1_0_0_1_1_1" "1_0_0_0_1_0_1_1_0_1_1"
## [58] "0_0_1_0_1_1_0_1_1_1_0" "1_0_0_0_0_0_1_1_0_0_0" "0_0_0_0_1_1_0_0_0_0_1"
```

Great, let's run then.

```
write.csv(x = df_doe2[,1:11],file="df_doe2.csv", row.names = F, quote = F)
```

Analysis

Let's get the corresponding results now (session has changed again...):

```
df=read.csv("Data",header=T)
str(df)
```

```
## 'data.frame':    66 obs. of  13 variables:
##  $ Date: chr  "2023-12-24-22:09:55" "2024-01-11-01:54:36" "2024-01-11-01:55:42" "2024-01-11-01:55:43
##  $ x1  : num  0.1 0.9 0.9 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x2  : num  0.2 0.4 0.4 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x3  : num  0.2 0.99 0.99 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x4  : num  0.3 0.9 0.9 0.99 0 0.5 0.5 0.5 0.5 0.5 ...
##  $ x5  : num  0.5 0.88 0.88 0.99 0.9 0.4 0.4 0.4 0.4 0.4 ...
##  $ x6  : num  0.6 0.44 0.44 0.99 0.9 0.5 0.5 0.5 0.5 0.5 ...
##  $ x7  : num  0.7 0.55 0.55 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
```

```
##  $ x8  : num  0.01 0.7 0.7 0.99 0.9 0.8 0.8 0.8 0.8 0.8 ...
##  $ x9  : num  0.4 0.8 0.8 0.99 0.9 0.3 0.3 0.3 0.3 0.3 ...
##  $ x10 : num  0.44 0.7 0.7 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x11 : num  0.6 0.9 0.9 0.99 0.9 0.6 0.6 0.6 0.6 0.6 ...
##  $ y   : num  0.477 0.359 0.356 -0.828 -0.599 ...
```

```
df = df[40:60,]
```

Now let's evaluate mean and variability for each combination

```
summary(aov(data=df, y~x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## x1           1  3.839   3.839   8.148 0.0135 *
## x3           1  1.583   1.583   3.360 0.0898 .
## x4           1  2.838   2.838   6.022 0.0290 *
## x6           1  0.011   0.011   0.023 0.8825
## x8           1  0.141   0.141   0.300 0.5930
## x10          1  0.437   0.437   0.927 0.3531
## x11          1  0.009   0.009   0.019 0.8916
## Residuals   13  6.125   0.471
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

All right, it is quite clear that only x1, x4, x7, and x9 have a real influence. This should help!

Guessing a model

DoE

Space filling designs are generally good for guessing a model. I only have 4 real dimensions, so maybe 40 experiments will be enough. Let's start with a purely uniform design and replace our four columns by a better design.

```
library(DiceDesign)
```

```
## Warning: package 'DiceDesign' was built under R version 4.3.2
```

```
set.seed(34234)
space_dim = 11
n_sample = 40
n_replicates = 1
x_init = runif(n=space_dim * n_sample *n_replicates, min=0, max=1)
df_doe3 = as.data.frame(matrix(data=x_init, ncol = space_dim))
names(df_doe3)=paste0("x",1:11)
df_doe3_lhs = lhsDesign(n_sample, 4, randomized=TRUE, seed=126982)
df_doe3_lhs = as.data.frame(df_doe3_lhs$design)
df_doe3[,c(1,4,7,9)]=df_doe3_lhs[,1:4]
df_doe3
```

```
##            x1         x2         x3          x4           x5         x6
## 1  0.16237596 0.56219857 0.31247555 0.483512231 0.4214900832 0.01103175
## 2  0.29372375 0.03584398 0.16454138 0.127640132 0.5650408522 0.18056113
## 3  0.95586374 0.65586362 0.84597806 0.623599293 0.7923115003 0.10185522
## 4  0.11544308 0.65541362 0.74552773 0.296935291 0.6934544423 0.08787425
## 5  0.58086844 0.88804161 0.60044120 0.362472244 0.9040959824 0.78772804
## 6  0.41499493 0.03151619 0.50319542 0.378200985 0.0003432578 0.81543241
## 7  0.06960270 0.54635519 0.17951550 0.693113987 0.3029135461 0.79521334
## 8  0.03253576 0.62423715 0.12626720 0.950143169 0.7090219986 0.99148585
## 9  0.08831747 0.84668117 0.01657084 0.093677943 0.3300319051 0.78532109
## 10 0.61478454 0.52400603 0.33273237 0.209304708 0.1436865469 0.37128190
## 11 0.79566713 0.67424949 0.96527611 0.853276060 0.4015820019 0.90282130
## 12 0.66090451 0.05259673 0.71072727 0.848302028 0.7727583568 0.98815454
## 13 0.90657664 0.80994245 0.80501681 0.928547122 0.3807693222 0.58569584
## 14 0.97678643 0.35834031 0.69907352 0.876108095 0.2860963887 0.23920572
## 15 0.82955517 0.63124463 0.09251739 0.627165315 0.5391579669 0.56627325
## 16 0.75497930 0.48317934 0.77918569 0.245505646 0.8999879120 0.70279341
## 17 0.22213316 0.86404293 0.41969023 0.114468456 0.1324844793 0.96967653
## 18 0.01736780 0.95072814 0.49308341 0.031886748 0.7398800019 0.50952782
## 19 0.12634060 0.34251639 0.93538180 0.348897687 0.8568507475 0.63930565
## 20 0.46696477 0.95754391 0.84351061 0.008213994 0.4602669028 0.34450980
## 21 0.26603610 0.50953563 0.07781781 0.541246296 0.8638894316 0.60246259
## 22 0.93366472 0.36392688 0.55273942 0.432011193 0.7594301109 0.91432180
## 23 0.37844606 0.77056833 0.93439293 0.654353600 0.1300427585 0.98831977
## 24 0.73050386 0.93735920 0.96059840 0.821403398 0.0716901077 0.81824874
## 25 0.44482804 0.20410667 0.37967489 0.178772649 0.4497509825 0.18919312
## 26 0.24523713 0.80567247 0.11430129 0.580809195 0.9702033254 0.50269653
## 27 0.55624674 0.66768172 0.01567389 0.070136938 0.3950183359 0.02306972
## 28 0.53386441 0.91523179 0.21539756 0.791094195 0.7334943563 0.56147165
## 29 0.19603595 0.21751807 0.85131629 0.704644586 0.7961060463 0.85378318
## 30 0.37041099 0.59676384 0.05163280 0.156308654 0.2194646234 0.88742068
## 31 0.51317721 0.65155722 0.25951714 0.413781695 0.5886331913 0.23958330
## 32 0.62921835 0.40840043 0.10786923 0.749560007 0.7300296347 0.23685063
## 33 0.48948764 0.40856151 0.52380592 0.567499365 0.4395907158 0.83770867
## 34 0.86296091 0.83605536 0.46377968 0.273427868 0.5540510407 0.73320794
## 35 0.80331737 0.01885588 0.03708099 0.459524006 0.0221745821 0.17996160
## 36 0.31498031 0.35238081 0.13796519 0.990472377 0.1161345481 0.59948580
## 37 0.34903384 0.20533362 0.81685214 0.311172770 0.7453484447 0.16316373
## 38 0.89257094 0.88693776 0.28166002 0.758934224 0.8965787503 0.51978081
## 39 0.69466368 0.92637289 0.62399203 0.903574330 0.5181920759 0.70332018
## 40 0.70504841 0.14817216 0.40887317 0.508784679 0.6713589109 0.24235704
##            x7         x8         x9         x10        x11
## 1  0.36493259 0.88305279 0.89522480 0.648386413 0.81480663
## 2  0.11219079 0.73967936 0.98839271 0.226001854 0.02913616
## 3  0.06372065 0.89536990 0.12223628 0.410500118 0.10396365
## 4  0.53558066 0.99833911 0.69894151 0.494595446 0.49804329
## 5  0.44536177 0.36692313 0.16800127 0.167479875 0.25653696
## 6  0.02180762 0.91250810 0.66306573 0.523623152 0.50324196
## 7  0.90247458 0.06233767 0.80308363 0.498943956 0.45838615
## 8  0.62890353 0.85124279 0.48229629 0.407975660 0.68465511
## 9  0.17536151 0.78795469 0.51815631 0.844492154 0.20580736
## 10 0.74315472 0.38080827 0.14650253 0.047554802 0.80482503
## 11 0.71507267 0.95475808 0.38437776 0.044056677 0.02100704
## 12 0.55181675 0.49502146 0.37225015 0.361021403 0.43915211
```

```
## 13 0.96373234 0.57234969 0.32565654 0.946051903 0.03960744
## 14 0.86205546 0.29516190 0.86481332 0.540710679 0.67216729
## 15 0.58696479 0.56908656 0.17921913 0.811452496 0.64926278
## 16 0.97664934 0.61549641 0.28805618 0.541803096 0.69730299
## 17 0.87948242 0.48991583 0.02051451 0.491588652 0.95601676
## 18 0.16365766 0.78121925 0.72865913 0.885190701 0.22860169
## 19 0.21162888 0.19307116 0.09315874 0.753740973 0.78618757
## 20 0.29237370 0.81910327 0.97484043 0.769474192 0.93741556
## 21 0.79723808 0.35697439 0.45016563 0.975948314 0.20853935
## 22 0.25970016 0.34103935 0.24191625 0.919056801 0.73202387
## 23 0.38818201 0.49444893 0.41962678 0.046190232 0.09916626
## 24 0.40726166 0.95880017 0.26084180 0.438154176 0.54723250
## 25 0.52296430 0.05469705 0.91899155 0.078736137 0.81992071
## 26 0.24606549 0.91729076 0.79537063 0.001141399 0.01614837
## 27 0.08930558 0.41036089 0.43664771 0.412796476 0.07498529
## 28 0.83659868 0.14731879 0.71359735 0.692137080 0.99877572
## 29 0.34780491 0.03807231 0.53976155 0.189089298 0.70211511
## 30 0.75487730 0.79933910 0.58049111 0.308025227 0.14092333
## 31 0.61431060 0.25989638 0.63902881 0.537739653 0.66375098
## 32 0.02828736 0.12239807 0.20790249 0.940231339 0.14968431
## 33 0.49602525 0.79202671 0.83206915 0.847043667 0.82472302
## 34 0.30739142 0.87728648 0.30742974 0.718482192 0.37627872
## 35 0.69044035 0.20492491 0.04684838 0.005081756 0.52452764
## 36 0.13862024 0.20639560 0.56373910 0.635857667 0.07050156
## 37 0.94127276 0.25312684 0.94125418 0.386161443 0.57748778
## 38 0.65181155 0.57626329 0.05630583 0.653616616 0.37597897
## 39 0.80282175 0.34714720 0.76006327 0.149224033 0.47121644
## 40 0.46853967 0.55867814 0.60516949 0.666145667 0.50460932
```

```
write.csv(x = df_doe3,file="df_doe3.csv", row.names = F, quote = F)
```

Analysis

Let's get the corresponding results now (session has changed again...):

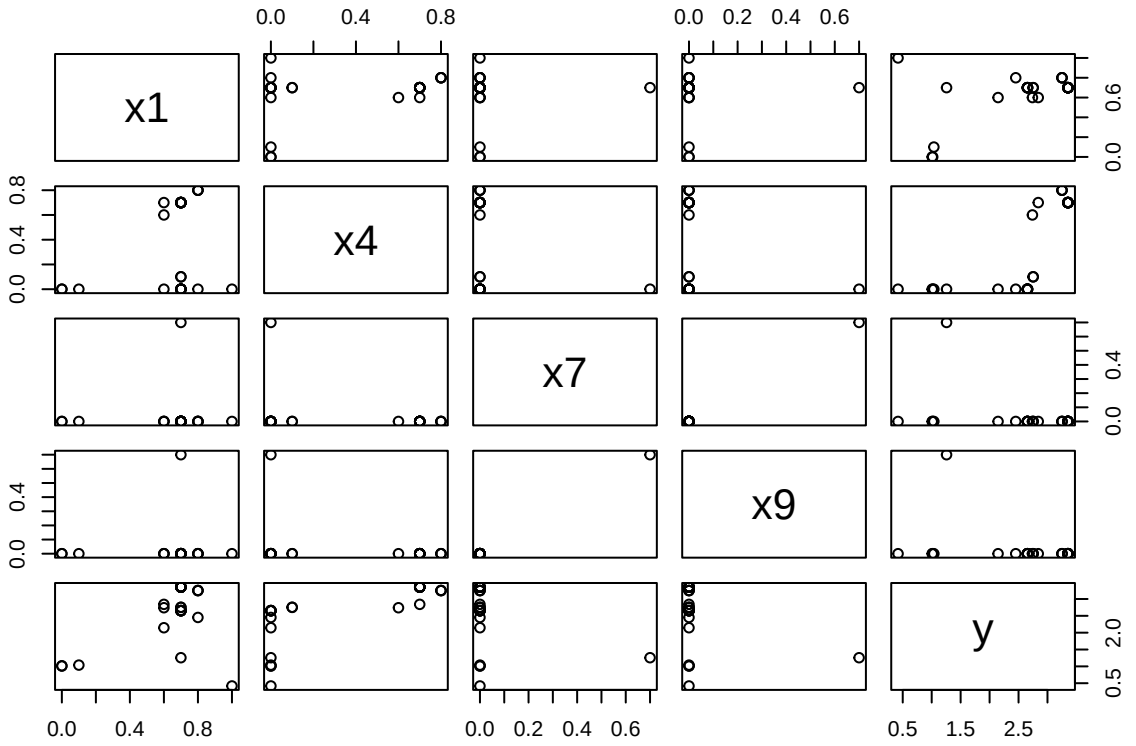
```
df=read.csv("Data",header=T)
str(df)
```

```
## 'data.frame':    66 obs. of  13 variables:
##  $ Date: chr  "2023-12-24-22:09:55" "2024-01-11-01:54:36" "2024-01-11-01:55:42" "2024-01-11-01:55:43
##  $ x1  : num  0.1 0.9 0.9 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x2  : num  0.2 0.4 0.4 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x3  : num  0.2 0.99 0.99 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x4  : num  0.3 0.9 0.9 0.99 0 0.5 0.5 0.5 0.5 0.5 ...
##  $ x5  : num  0.5 0.88 0.88 0.99 0.9 0.4 0.4 0.4 0.4 0.4 ...
##  $ x6  : num  0.6 0.44 0.44 0.99 0.9 0.5 0.5 0.5 0.5 0.5 ...
##  $ x7  : num  0.7 0.55 0.55 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x8  : num  0.01 0.7 0.7 0.99 0.9 0.8 0.8 0.8 0.8 0.8 ...
##  $ x9  : num  0.4 0.8 0.8 0.99 0.9 0.3 0.3 0.3 0.3 0.3 ...
##  $ x10 : num  0.44 0.7 0.7 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x11 : num  0.6 0.9 0.9 0.99 0.9 0.6 0.6 0.6 0.6 0.6 ...
##  $ y   : num  0.477 0.359 0.356 -0.828 -0.599 ...
```

```
df = df[30:60,]
```

We're interested in the last row:

```
df %>% select(-Date) -> df
plot(df[,c(1,4,7,9,12)])
```



Let's plot this more nicely:

```
ggplot(df, aes(y=y, x=x1)) + geom_point() + geom_smooth() + geom_smooth(method = "lm", formula = y ~ po
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 1.005
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 0.305
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 0
```

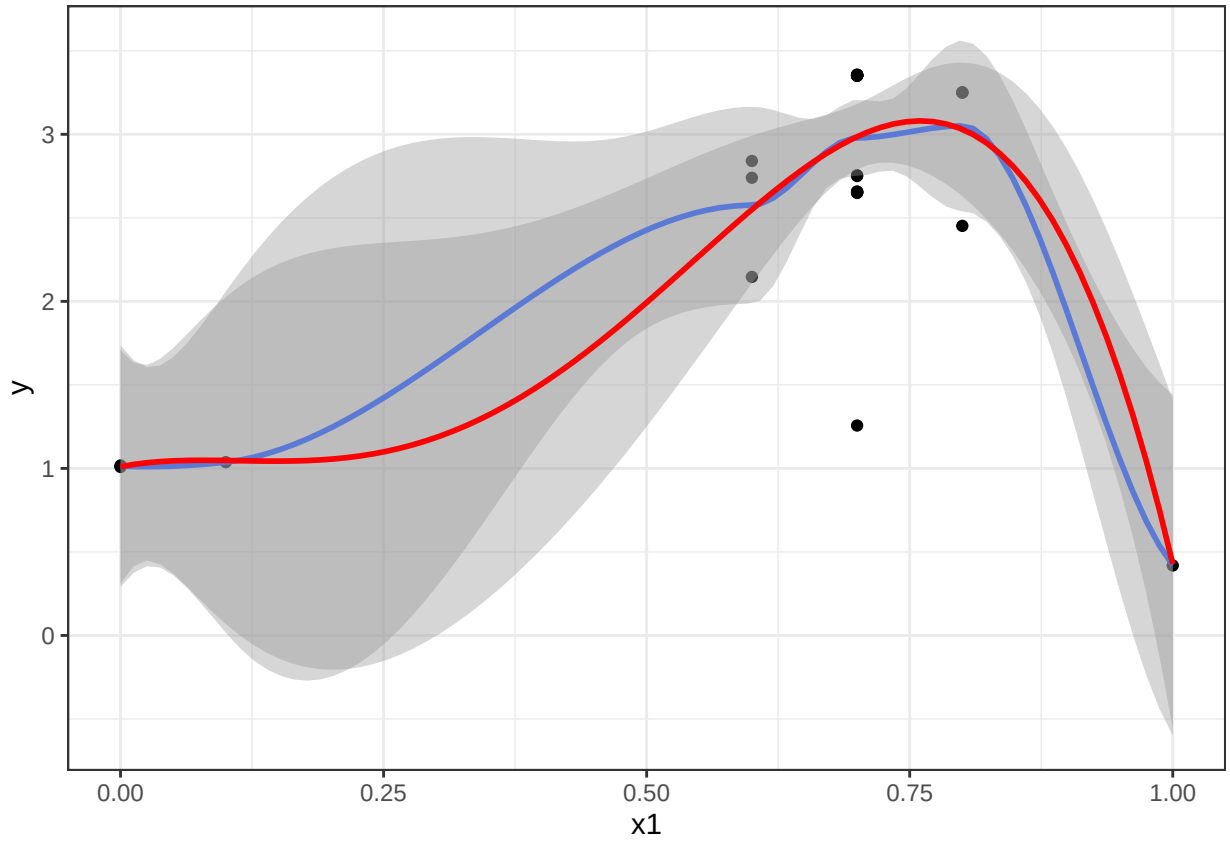
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 0.01
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at
## 1.005
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 0.305
```

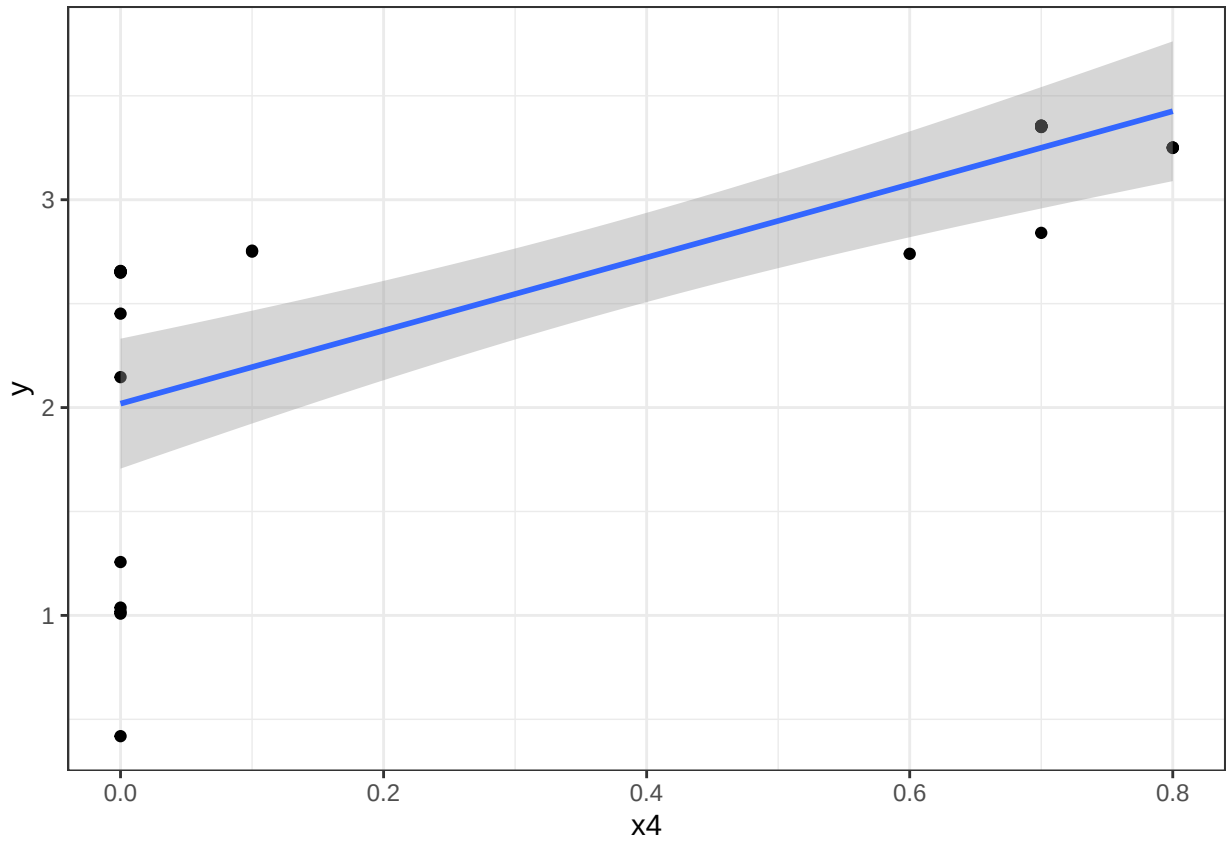
```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 0.01
```



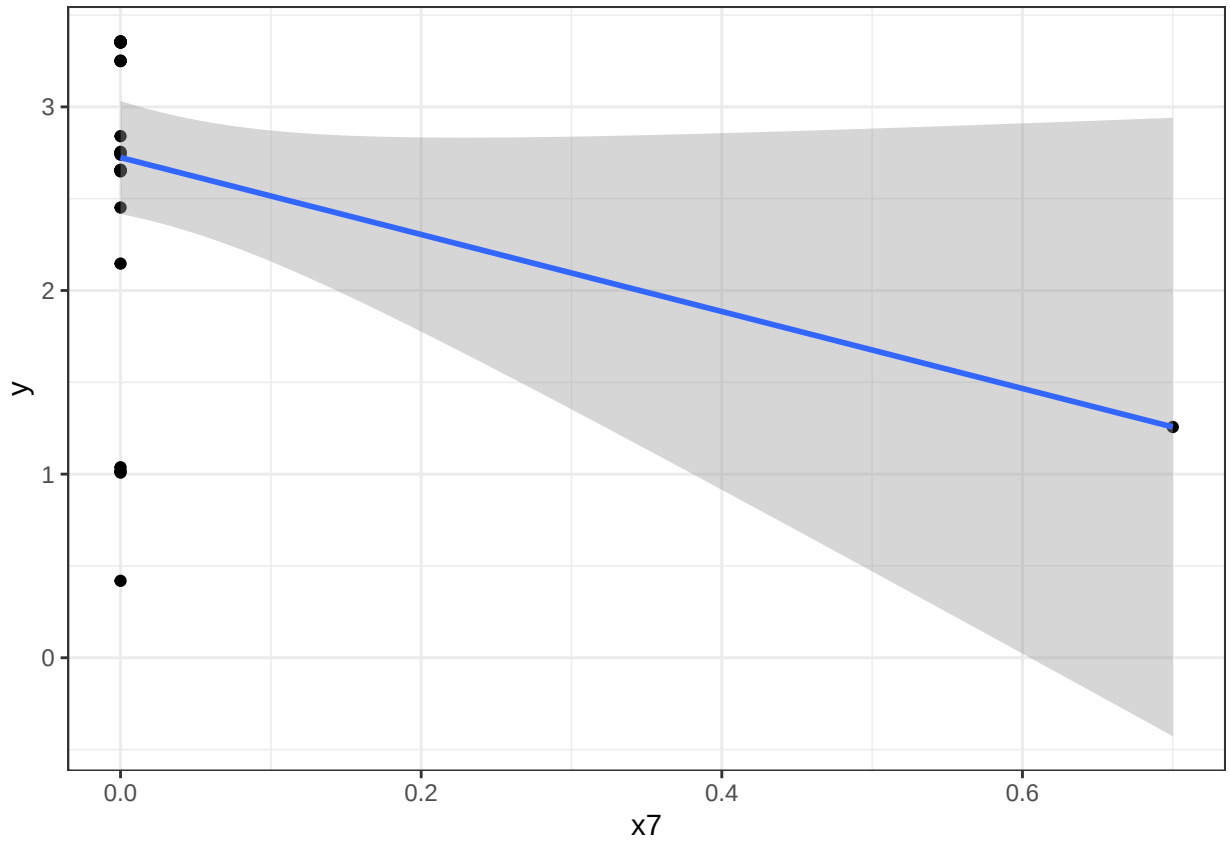

```
ggplot(df, aes(y=y, x=x4)) + geom_point() + geom_smooth(method="lm") + theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



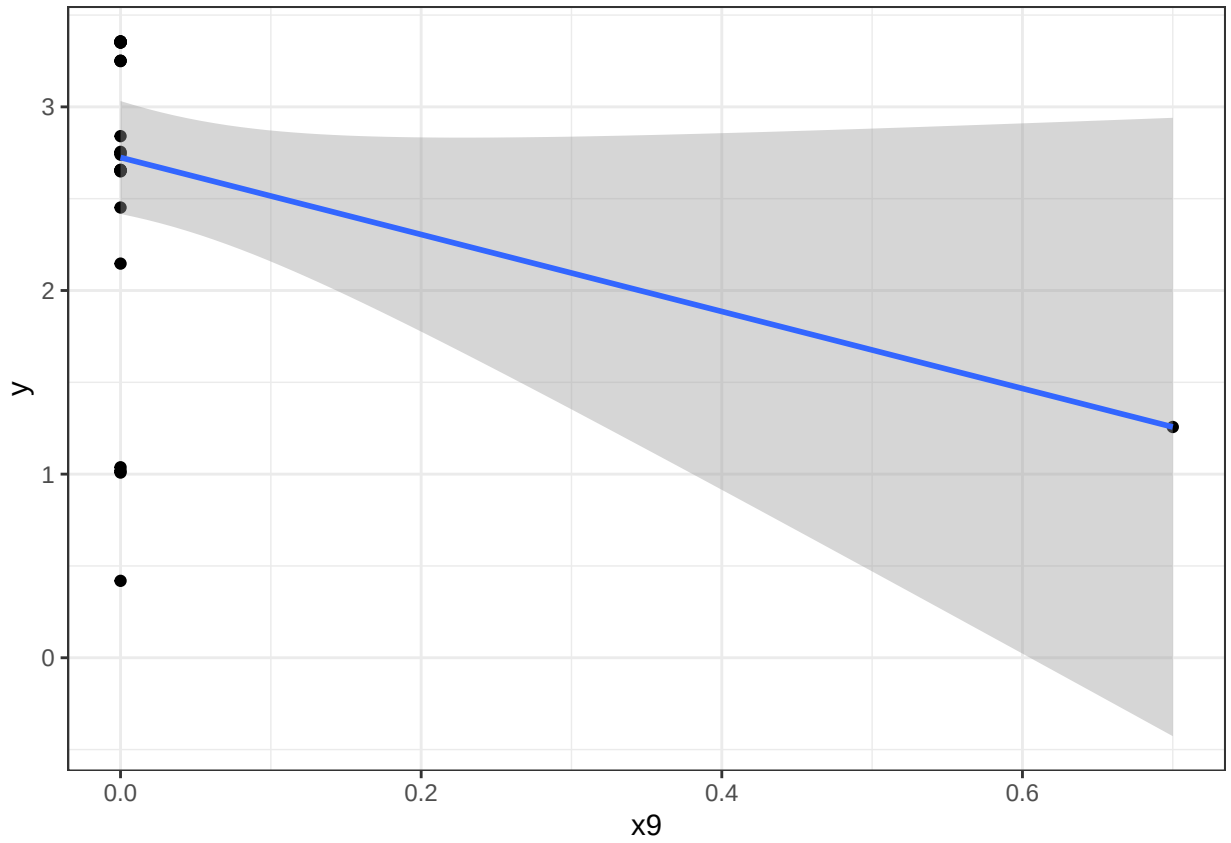

```
ggplot(df, aes(y=y, x=x7)) + geom_point() + geom_smooth(method="lm") + theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
ggplot(df, aes(y=y, x=x9)) + geom_point() + geom_smooth(method="lm") + theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



All right, unlike x1, which is strongly non linear, x4, x7, and x9 could be considered as such. x9 a a clear negative slope but it's not so clear for x4 and x7. Let's run a close model (a `poly(x1,4)` is obviously biased, just like a loess, but it will do) for this:

```
summary(lm(data=df, y~poly(x1,4) + x4 + x7 + x9))
```

```
##
## Call:
## lm.default(formula = y ~ poly(x1, 4) + x4 + x7 + x9, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.16556 -0.01607  0.02856  0.05701  0.06148
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.36402    0.02482  95.228  < 2e-16 ***
## poly(x1, 4)1  1.85612    0.09104  20.388  < 2e-16 ***
## poly(x1, 4)2 -2.34693    0.08847 -26.527  < 2e-16 ***
## poly(x1, 4)3 -1.37137    0.08722 -15.723 3.88e-14 ***
## poly(x1, 4)4 -0.48834    0.08634  -5.656 7.99e-06 ***
## x4            0.95443    0.04958  19.250 4.25e-16 ***
## x7           -1.95790    0.12889 -15.190 8.25e-14 ***
## x9                NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.0857 on 24 degrees of freedom
## Multiple R-squared:  0.9919, Adjusted R-squared:  0.9899
## F-statistic: 489.3 on 6 and 24 DF,  p-value: < 2.2e-16
```

All right, the effect of x7 is really not so clear but the rest is not so bad. x1 seems to explain most of the variability (the remaining variability around x1 is rather low compared to the one coming from x1) although x4, x7, and x9 appear to contribute to the rest. Since we're looking for the higher value, setting x9 to 0 appears like a good choice. For x4, which is not as strong, setting it to 1 also appears like a good choice, and for x7, it is not so clear. Anyway, sampling in the $[0.6, 0.85]$ range for x1, is a safe choice.

Looking for an optimal configuration

DoE

```
set.seed(234981)
space_dim = 11
n_sample = 40
n_replicates = 1
x_init = runif(n=space_dim * n_sample *n_replicates, min=0, max=1)
df_doe4 = as.data.frame(matrix(data=x_init, ncol = space_dim))
names(df_doe4)=paste0("x",1:11)
df_doe4$x1 = runif(n = n_sample, min=.6, max=.85)
df_doe4$x9 = 1 # I made a mistake here! x9 should have been a 0!
df_doe4$x4 = 1
write.csv(x = df_doe4,file="df_doe4.csv", row.names = F, quote = F)
```

Analysis

Let's get the corresponding results now (session has changed again...):

```
df=read.csv("Data",header=T)
str(df)
```

```
## 'data.frame':    66 obs. of  13 variables:
##  $ Date: chr  "2023-12-24-22:09:55" "2024-01-11-01:54:36" "2024-01-11-01:55:42" "2024-01-11-01:55:43
##  $ x1  : num  0.1 0.9 0.9 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x2  : num  0.2 0.4 0.4 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x3  : num  0.2 0.99 0.99 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x4  : num  0.3 0.9 0.9 0.99 0 0.5 0.5 0.5 0.5 0.5 ...
##  $ x5  : num  0.5 0.88 0.88 0.99 0.9 0.4 0.4 0.4 0.4 0.4 ...
##  $ x6  : num  0.6 0.44 0.44 0.99 0.9 0.5 0.5 0.5 0.5 0.5 ...
##  $ x7  : num  0.7 0.55 0.55 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x8  : num  0.01 0.7 0.7 0.99 0.9 0.8 0.8 0.8 0.8 0.8 ...
##  $ x9  : num  0.4 0.8 0.8 0.99 0.9 0.3 0.3 0.3 0.3 0.3 ...
##  $ x10 : num  0.44 0.7 0.7 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x11 : num  0.6 0.9 0.9 0.99 0.9 0.6 0.6 0.6 0.6 0.6 ...
##  $ y   : num  0.477 0.359 0.356 -0.828 -0.599 ...
```

```r
df = df[34:66,]
df %>% select(-Date) -> df
```

Let's plot again:

```r
ggplot(df, aes(y=y, x=x1)) + geom_point() + geom_smooth() + geom_smooth(method = "lm", formula = y ~ po
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 0.8
```

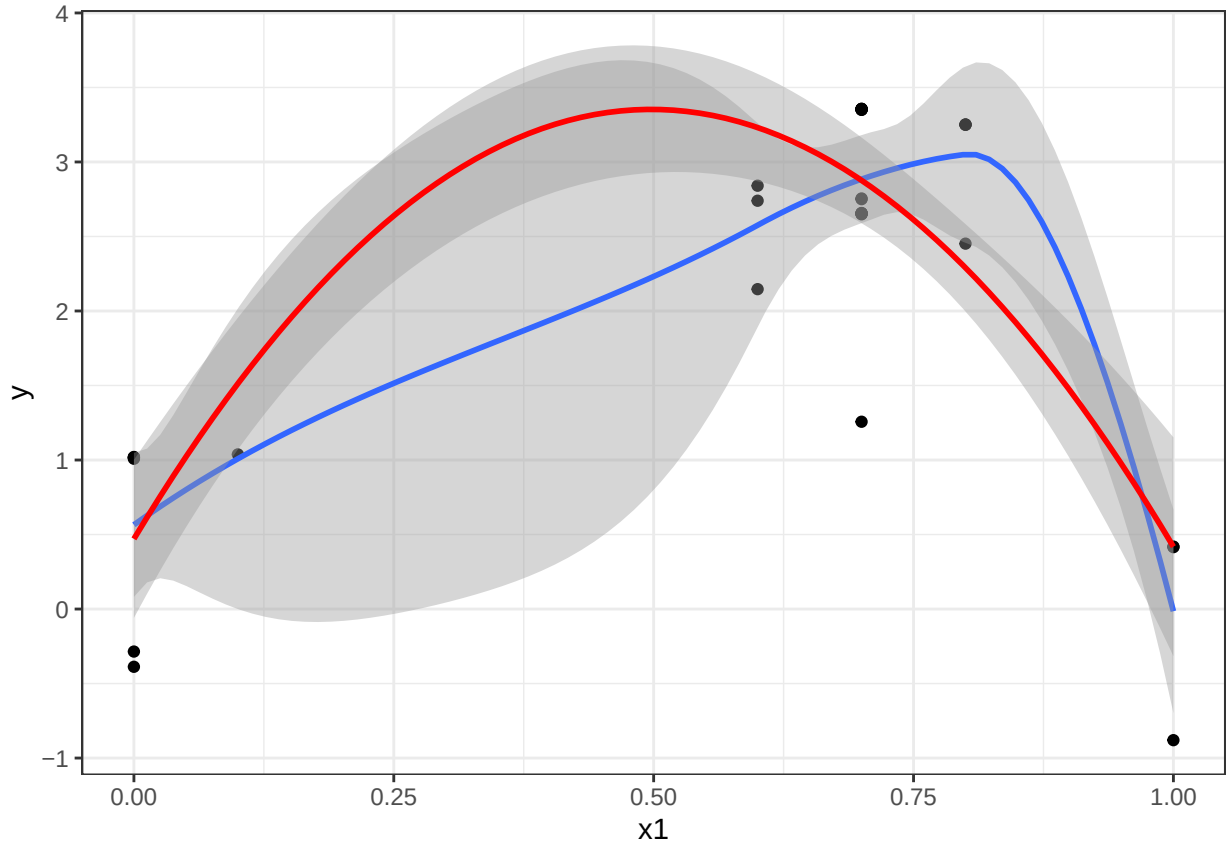
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 0.2
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 9.0749e-17
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at
## 0.8
```

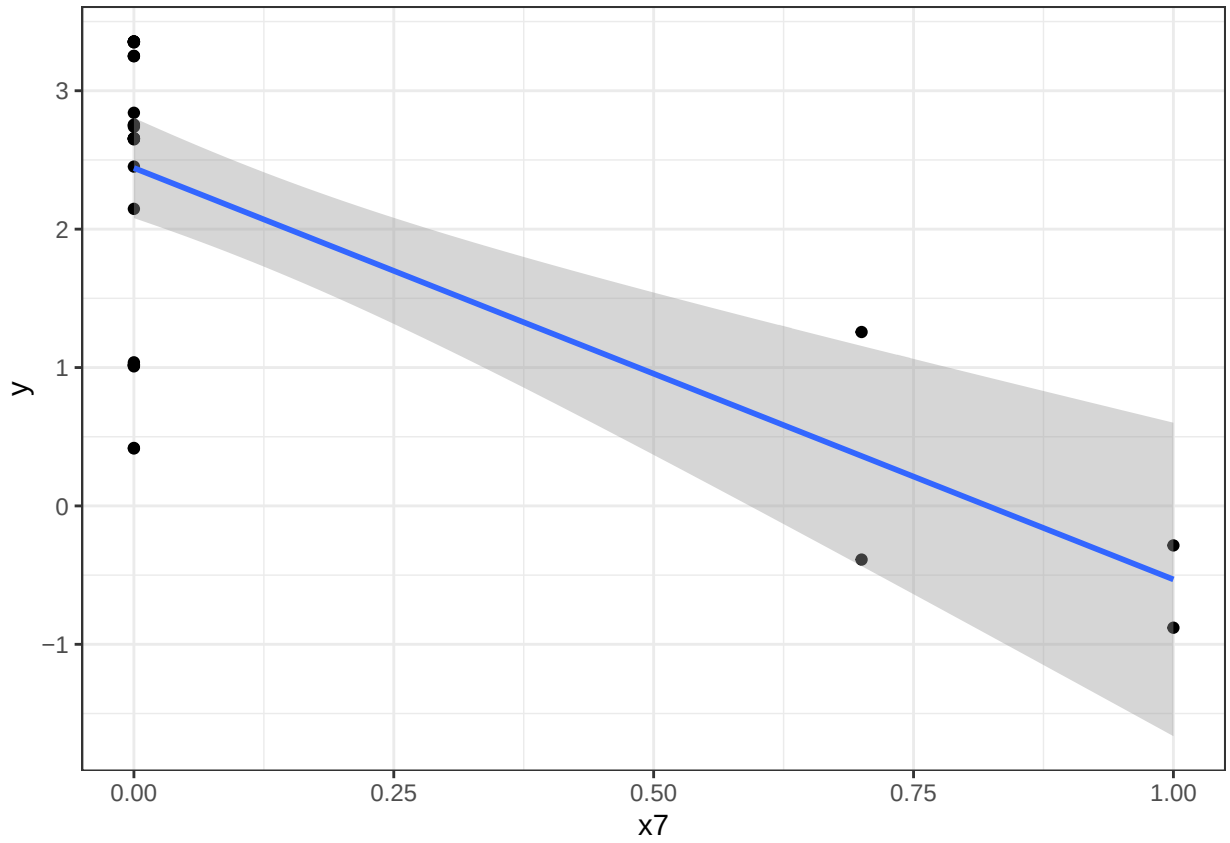
```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 0.2
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 9.0749e-17
```




```
ggplot(df, aes(y=y, x=x7)) + geom_point() + geom_smooth(method="lm") + theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



All right, now, it is pretty clear that setting x4 to 1 would be a good idea in this neighborhood.

The optimal configuration is thus - $x1 \approx 0.72$ - $x4 = 1$ - $x7 = 1$ - $x9 = 1$. oh Wait! 0! And that's why y is not close to 3 anymore! All other parameters are of no importance

DoE2

One more time!

```
set.seed(2981)
space_dim = 11
n_sample = 20
n_replicates = 1
x_init = runif(n=space_dim * n_sample *n_replicates, min=0, max=1)
df_doe5 = as.data.frame(matrix(data=x_init, ncol = space_dim))
names(df_doe5)=paste0("x",1:11)
df_doe5$x1 = runif(n = n_sample, min=.68, max=.78)
df_doe5$x9 = 0
df_doe5$x4 = 1
df_doe5$x7 = 1
write.csv(x = df_doe5,file="df_doe6.csv", row.names = F, quote = F)
```

Analysis

Let's get the corresponding results now (session has changed again...):

```r
df=read.csv("Data",header=T)
str(df)
```

```
## 'data.frame':    66 obs. of  13 variables:
##  $ Date: chr  "2023-12-24-22:09:55" "2024-01-11-01:54:36" "2024-01-11-01:55:42" "2024-01-11-01:55:43
##  $ x1  : num  0.1 0.9 0.9 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x2  : num  0.2 0.4 0.4 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x3  : num  0.2 0.99 0.99 0.99 0.9 0.4 0.5 0.6 0.8 0.9 ...
##  $ x4  : num  0.3 0.9 0.9 0.99 0 0.5 0.5 0.5 0.5 0.5 ...
##  $ x5  : num  0.5 0.88 0.88 0.99 0.9 0.4 0.4 0.4 0.4 0.4 ...
##  $ x6  : num  0.6 0.44 0.44 0.99 0.9 0.5 0.5 0.5 0.5 0.5 ...
##  $ x7  : num  0.7 0.55 0.55 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x8  : num  0.01 0.7 0.7 0.99 0.9 0.8 0.8 0.8 0.8 0.8 ...
##  $ x9  : num  0.4 0.8 0.8 0.99 0.9 0.3 0.3 0.3 0.3 0.3 ...
##  $ x10 : num  0.44 0.7 0.7 0.99 0.9 0.7 0.7 0.7 0.7 0.7 ...
##  $ x11 : num  0.6 0.9 0.9 0.99 0.9 0.6 0.6 0.6 0.6 0.6 ...
##  $ y   : num  0.477 0.359 0.356 -0.828 -0.599 ...
```

```r
# df = df[381:400,]
df = df[30:66,]
df %>% select(-Date) -> df
```

Let's plot again:

```r
ggplot(df, aes(y=y, x=x1)) + geom_point() + geom_smooth() + geom_smooth(method = "lm", formula = y ~ po
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 1.005
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 0.305
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 5.5338e-17
```

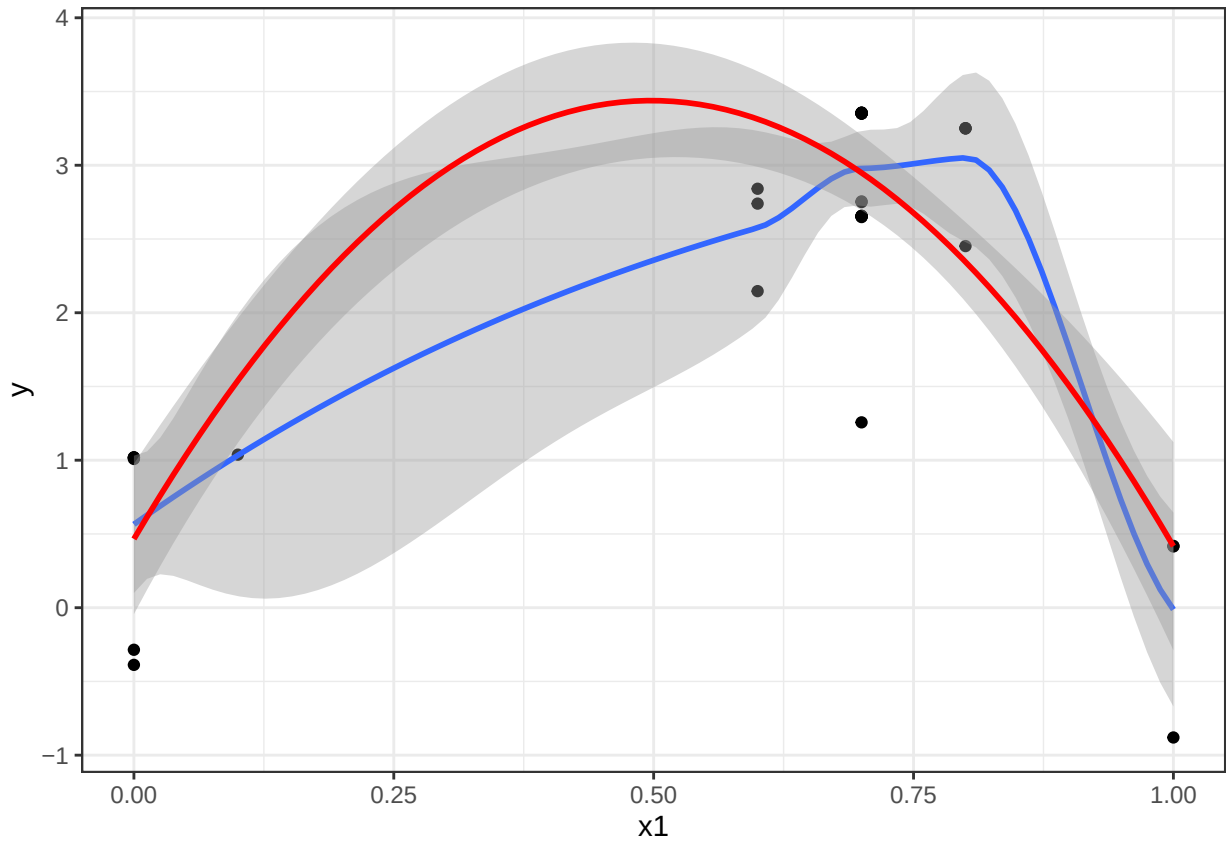
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 0.04
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at
## 1.005
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 0.305
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 5.5338e-17
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 0.04
```



Anyway, the optimal configuration is thus - $x1 \approx 0.73$ - $x4 = 1$ - $x7 = 1$ - $x9 = 9$. All other parameters are of no importance and the optimal value for y is around 3.42.

```
df=read.csv("Data",header=T)
df %>% select(-Date) -> df
```

```
df[df$y==max(df$y),]
```

```
##     x1 x2 x3  x4 x5  x6 x7  x8 x9 x10 x11        y
## 39 0.7  0  0 0.7  0 0.7  0 0.7  0 0.1 0.6 3.356876
```