Assignment4

Factors and levels

By factor we refer to variable that can be independently varied during the series of experiments. The possible values of factors (within the experiment series) is called as level.

- x1 = Arrival distribution (AD): Exp, Unif
- x2 = Average arrival rate (AA): 25, 22.5
- x3 = Preparation time (PT): exp(40) or Unif(30,50)
- x4 = Recovery time (RT): exp(40) or Unif(30,50)
- x5 = Preparation units: (PU) 4,5
- x6 = Recovery units (RU): 4,5

Test design

- "One factor at time" needs 7 experiments
- "Full experiment" needs 64 experiments
- We are aiming for 2^(6-3) = 8 experiments
- 3-way and higher level joint effects are assumed to 0
- 2-way joint effects that can be also assumed to be ~0: x1x2, x3x4 x5x6 With these settings test design matrix is following.

```
designMatrix = [
    [1,0,1,0,1,0],
    [1,0,1,0,0,1],
    [1,0,0,1,1,0],
    [1,0,0,1,0,1],
    [0,1,0,1,0,1],
    [0,1,0,1,1,0],
    [0,1,1,0,0,1],
    [0,1,1,0,1,0],
]
```

This matrix were achieved by setting all discarded 2-way intereactions to -1 as demonstrated in lectures.

- Initial regression model has a form:

y ~ x1 + x2 + x3 + x4 +x5 +x6 +x1*x3 + x1*x4 + x1*x5 + x1*x6 + x2*x3 + x2*x4 +x2*x5 + x2*x6 + x3*x5 + x3*x6 + x4*x5 + x4*x6

Test serial correlation

Autocorrelation analysis is made by plotting 10 independent samples autocorrelations of entrance queue lengths. Based on plots one can see that queues are autocorrelated at least 250 time steps (see when autocorrelations cross the 95 % confidence interval of no autocorrelation). If samples for regression analysis were made from one patient stream, the time interval between collecting samples should be at least 250 times steps.

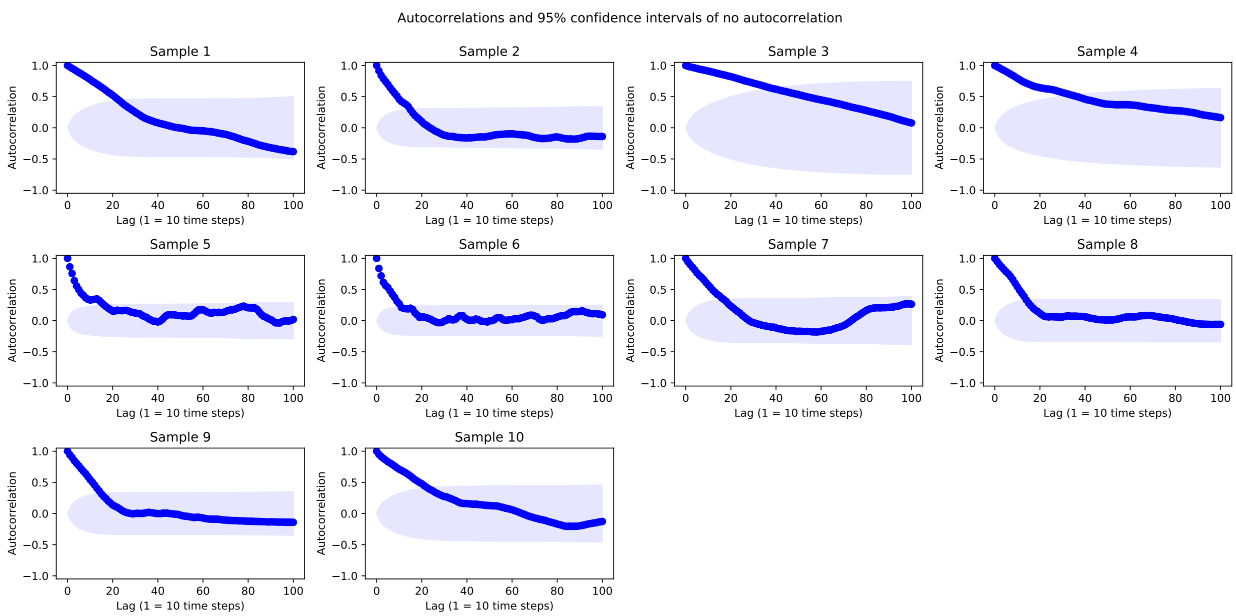
Python codes for the analysis is located:

https://github.com/RiskyRisto/simLekuri/tree/main/Python_project

```
random_seeds = [*range(settings.N_SAMPLES)]
print("AUTOCORRELATION ANALYSIS")
```

```
## AUTOCORRELATION ANALYSIS
```

```
samples = run_experiment(settings.CONFIGURATIONS[0], 10, True, None)
statistics.acf_plots(samples)
```



Regression model

Build a regression model for the average queue length. For each configuration multiple simulations were conducted and linear regression model was fitted using resulting "(n_samples * 8) rows, 6 columns" matrix and respective queue times. Number of replications, n_samples, were set to 10.

Newt we simulate the data and run an initial regression model. Python code is located:

https://github.com/RiskyRisto/simLekuri/tree/main/Python_project

```
print("SIMULATION FOR INDEPENDENT SAMPLES")
#print("-"*40)
```

```
run_simulation(True)
```

```
print("SIMULATION FOR DEPENDENT SAMPLES")
#print("-"*40)
```

```
run_simulation(False)
```

```
regression.experiment()
```

```
regression.experiment()
```

```
## --------------------
## REGRESSION
## --------------------
## coefficients [-1.03  1.03  0.16 -0.16 -0.24  0.24]
## intercept 1.3195891783567137
## prediction 1.320
## simulated 1.549
## mean squared error 1.893
## explained variance score 0.377
## r^2 score 0.377
## --------------------
```

Further analysis with R

- There are perfect dependecies within x variables causing singularities
- Parameters x2, x4 and x6 had to be dropped out of model because of singularity issues.
- Model to fit is: y ~ b0 + b1*x1 + b2*x3 + b3*x5 + b4*x1:x3 + b5*x1:x5 + b6*x3:x5 + eps
- Statistically significant b1, coefficient which corresponds to the arrival distribution.
- Statistically significant b5, coefficient which corresponds to the preparation rooms.
- Other coefficients are not statistically significant.

Correlation matrix of the variables:

```
dat <- read.csv("experiment_data.csv", header = FALSE)
names(dat) <- c("x1", "x2", "x3", "x4", "x5", "x6", "y")
print(round(cor(dat),2))
```

```
##       x1    x2    x3    x4    x5    x6     y
## x1  1.00 -1.00  0.00  0.00  0.00  0.00 -0.59
## x2 -1.00  1.00  0.00  0.00  0.00  0.00  0.59
## x3  0.00  0.00  1.00 -1.00  0.00  0.00  0.09
## x4  0.00  0.00 -1.00  1.00  0.00  0.00 -0.09
## x5  0.00  0.00  0.00  0.00  1.00 -1.00 -0.14
## x6  0.00  0.00  0.00  0.00 -1.00  1.00  0.14
## y  -0.59  0.59  0.09 -0.09 -0.14  0.14  1.00
```

```r
# Set x-varibles as factors
dat$x1 <- factor(dat$x1)
dat$x2 <- factor(dat$x2)
dat$x3 <- factor(dat$x3)
dat$x4 <- factor(dat$x4)
dat$x5 <- factor(dat$x5)
dat$x6 <- factor(dat$x6)

# Full model not defined because of singularities
fit <- lm(y ~ x1 + x3 + x5 + x1*x3 + x1*x5 + x3*x5, data = dat)

print(summary(fit))
```

```
##
## Call:
## lm(formula = y ~ x1 + x3 + x5 + x1 * x3 + x1 * x5 + x3 * x5,
##     data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0363 -0.4666 -0.1570  0.1523  6.3079
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.5493     0.4160   6.128  4.1e-08 ***
## x11          -2.0841     0.5447  -3.826 0.000272 ***
## x31           0.4714     0.5447   0.866 0.389556
## x51          -1.0983     0.5447  -2.016 0.047427 *
## x11:x31      -0.7525     0.6289  -1.197 0.235367
## x11:x51       0.7988     0.6289   1.270 0.208074
## x31:x51       0.4565     0.6289   0.726 0.470236
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.406 on 73 degrees of freedom
## Multiple R-squared:  0.4057, Adjusted R-squared:  0.3568
## F-statistic: 8.304 on 6 and 73 DF,  p-value: 7.264e-07
```

```r
# 95% confidence intervals for the coefficients
print(confint(fit))
```

```
##                  2.5 %      97.5 %
## (Intercept)  1.7202871  3.37841028
## x11         -3.1695645 -0.99857172
## x31         -0.6140535  1.55693930
## x51         -2.1837930 -0.01280018
## x11:x31     -2.0059283  0.50091828
## x11:x51     -0.4546257  2.05222089
## x31:x51     -0.7969103  1.70993632
```

Conclusions

When arrival distribution is uniform distribution, estimate for mean entrance queue length is -2.08 lower than when distribution is exponential. b5 corresponds to preparation rooms, when there are 5 rooms instead of 4 rooms, the estimated mean entrance queue length is -1.10 lower. This result makes sense.

Discussion

It seems that selecting design matrix, as demonstrated in lectures with 3-way joint effect, is not working with 2-way joint effects because it causes perfect dependence between variables.

If we would take samples for every configuration, there would not be singularity issues and results would be much more reliable and better.