Efficiency of OLS for linear models with correlated data

In [1]:

```
import collections
import numpy as np
import pandas as pd
from scipy import linalg
```

In [2]:

```
def make_covariates(n, cluster_covariates):
    covariates = np.tile(cluster_covariates, n)
    return np.column_stack((np.ones_like(covariates), covariates))
```

In [3]:

```
def make_exponential_correlation_matrix(m, rho):
    correlation_matrix = np.eye(m)
    for i in range(m - 1):
        for j in range(i + 1, m):
            correlation_matrix[i, j] = correlation_matrix [j, i] = np.power(rho, j
- i)
    return correlation_matrix
```

In [4]:

```
def compute_efficiency(n, cluster_covariates, rho):
    X = make_covariates(n, cluster_covariates)
    sigma = linalg.block_diag(*(
        [make_exponential_correlation_matrix(len(cluster_covariates), rho)]*n))

    gram_matrix_inv = linalg.cho_solve(linalg.cho_factor(X.T.dot(X)), np.eye(X.sha
pe[1]))
    covariance_ols = gram_matrix_inv.dot(X.T.dot(sigma).dot(X)).dot(gram_matrix_in
v)

    weights = linalg.cho_solve(linalg.cho_factor(sigma), np.eye(X.shape[0]))
    covariance_gls = linalg.cho_solve(
        linalg.cho_factor(X.T.dot(weights).dot(X)), np.eye(X.shape[1]))

    return np.diag(covariance_gls)/np.diag(covariance_ols)
```

In [5]:

```
N = 10
CLUSTER_COVARIATES = [
    [-2,-1,0,1,2],
    [-1,-2,0,2,1],
    [0,-1,1,3,2],
    [0,-1,1,5,2],
]
RHO = np.hstack((np.linspace(0.1, 0.9, 9), [0.99]))

efficiency_results = collections.OrderedDict([
    (
        str(cluster_covariates),
        {
            str(np.round(rho, 2)): compute_efficiency(N, cluster_covariates, rho)
            for rho in RHO
        },
    )
    for cluster_covariates in CLUSTER_COVARIATES
])
```

In [6]:

```
efficiency_table = pd.DataFrame(
    index=pd.MultiIndex.from_product([
        map(str, CLUSTER_COVARIATES),
        ['$e(\\hat{\\beta}_0)$', '$e(\\hat{\\beta}_1)$'],
    ], names=['$x$', 'Value']),
    columns=pd.Series(map(lambda rho: str(np.round(rho, 2)), RHO), name='$\\rho$'
),
)
for cluster_covariates, values in efficiency_results.items():
    for rho, efficiencies in values.items():
        efficiency_table[rho][cluster_covariates, '$e(\\hat{\\beta}_0)$'] = effici
encies[0]
        efficiency_table[rho][cluster_covariates, '$e(\\hat{\\beta}_1)$'] = effici
encies[1]

with open('p2_efficiencies.tex', 'w') as f:
    f.write(efficiency_table.to_latex(
        escape=False,
        float_format=lambda f: str(np.round(f, 4))).replace('[', '(').replace(']',
')'))

efficiency_table
```

Out[6]:

```
             ρ       0.1       0.2       0.3       0.4       0.5       0.6       0.7       0.8       0.9     0.9
     x   Value
[-2,    e(β̂₀)    0.99776  0.991707  0.982965  0.972888  0.963082  0.955424  0.952117  0.955846  0.970123  0.99611
 -1,
  0,
  1,    e(β̂₁)   0.996874    0.9893  0.979685  0.969955  0.961538  0.955424  0.952221  0.952221  0.955424  0.96081
  2]
[-1,    e(β̂₀)    0.99776  0.991707  0.982965  0.972888  0.963082  0.955424  0.952117  0.955846  0.970123  0.99611
 -2,
  0,
  2,    e(β̂₁)   0.995921   0.98184  0.955424  0.915402  0.862069  0.797438  0.724923  0.648636  0.572575  0.50702
  1]
 [0,    e(β̂₀)   0.997183  0.988849  0.975751  0.959626  0.943244  0.930221  0.924682   0.93103  0.954102  0.99415
 -1,
  1,
  3,    e(β̂₁)   0.995921   0.98184  0.955424  0.915402  0.862069  0.797438  0.724923  0.648636  0.572575  0.50702
  2]
 [0,    e(β̂₀)   0.994938  0.981695   0.96362  0.944474  0.928059  0.917849   0.91686  0.927875  0.954054  0.99432
 -1,
  1,
  5,    e(β̂₁)   0.991123  0.964352  0.920638  0.862553  0.793975  0.719406  0.643193  0.568935  0.499199  0.44157
  2]
```