МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»
(Университет ИТМО)

Факультет систем управления и робототехники

# ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

по дисциплине

*«Теория оптимального управления»*

Студент:
*Группа № R3435* *Зыкин Л. В.*

Предподаватель:
*ведущий научный сотрудник, доцент* *Парамонов А. В.*

Санкт-Петербург
2025

# 1 ЛАБОРАТОРНАЯ РАБОТА №3. LQR ДЛЯ ЛИНЕЙНОГО СТАЦИОНАРНОГО ОБЪЕКТА

**Постановка (вариант 13)**

Дана система $\dot{x} = Ax + bu$, матрицы

$$A = \begin{bmatrix} 0 & 1 \\ 4 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ 6 \end{bmatrix}, \quad Q = \begin{bmatrix} 6 & 0 \\ 0 & 3 \end{bmatrix}, \quad r = 2.$$

Необходимо:

1. Найти оптимальный регулятор $u = -Kx$ на основе алгебраического уравнения Риккати (CARE)

$$A^T P + PA - Pbr^{-1}b^T P + Q = 0, \qquad K = r^{-1}b^T P.$$

2. Смоделировать замкнутую систему $\dot{x} = (A - bK)x$ при $x(0) = [1, 0]^T$, построить графики $x_1, x_2, u, J(t) = \int_0^t (x^T Q x + ru^2) d\tau$, найти установившееся значение $J(\infty)$.
3. Незначительно изменить $K$ (сохраняя устойчивость) и сравнить с оптимальным по критерию $J$.
4. Повторить моделирование для трёх значений $r > 0$ и $Q_k = kQ$ ($k > 0$), где для одного случая $Q_k = Q$.

## 1.1 Решение CARE и формула регулятора

Матрица $br^{-1}b^T = (1/r)\, bb^T$. Решая CARE для $P \succ 0$, получаем $K = r^{-1}b^T P$. Далее проверяем устойчивость матрицы $A - bK$ (собственные значения в левой полуплоскости). Численные значения приводятся ниже и воспроизводятся скриптом.

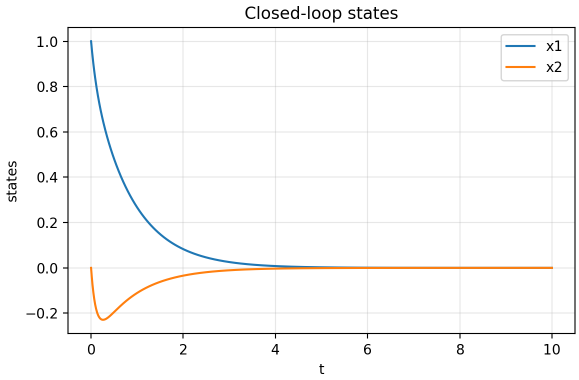
### 1.1.1 Численная реализация

Листинг 1.1 — LQR (вариант 13): решение CARE, моделирование и графики

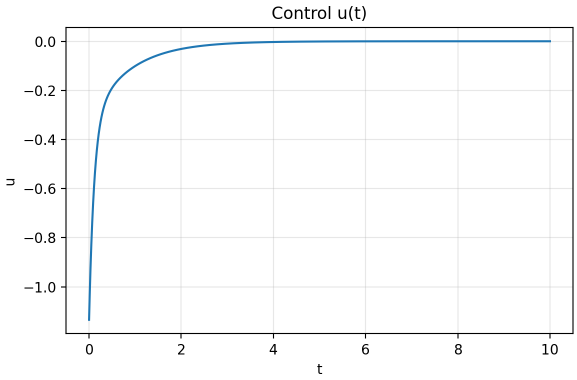
```
# lab3/python/lqr_var13.py
```

```
import numpy as np
from scipy.linalg import solve_continuous_are, eigvals
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt

A = np.array([[0.0, 1.0],[4.0, 3.0]])
b = np.array([[2.0],[6.0]])
Q = np.array([[6.0, 0.0],[0.0, 3.0]])
r = 2.0

# CARE
P = solve_continuous_are(A, b, Q, r)
K = (1.0/r) * (b.T @ P)  # shape (1,2)
Acl = A - b @ K
print("K=", K)
print("eig(A-bK)=", eigvals(Acl))

# Cost integrand
def lqr_cost(x, u):
    return float(x.T @ Q @ x + r * u**2)

# Closed-loop dynamics and online cost accumulation
x0 = np.array([1.0, 0.0])
T = (0.0, 10.0)

def odefun(t, z):
    x = z[:2]
    u = - float(K @ x)
    jdot = lqr_cost(x, u)
    dx = (Acl @ x)
    return np.hstack([dx, jdot])

z0 = np.hstack([x0, 0.0])
sol = solve_ivp(odefun, T, z0, max_step=0.01, rtol=1e-8, atol=1e
   -10)

t = sol.t
x = sol.y[:2,:]
Jt = sol.y[2,:]
u = - (K @ x).ravel()

import os
```

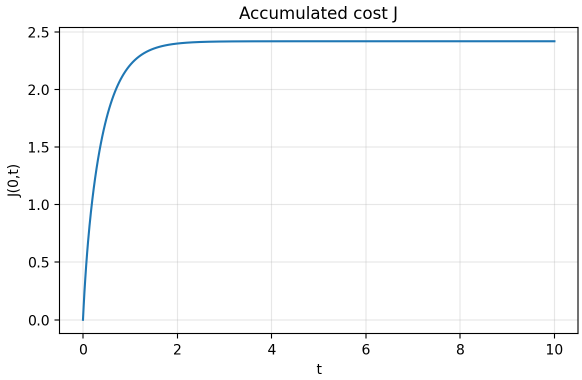
```
os.makedirs('/home/leonidas/projects/itmo/optimal-control-theory/
   lab3/images/task3', exist_ok=True)

plt.figure(figsize=(6,4))
plt.plot(t, x[0], label='x1')
plt.plot(t, x[1], label='x2')
plt.xlabel('t'); plt.ylabel('states'); plt.grid(True, alpha=0.3)
plt.title('Closed-loop states')
plt.legend(); plt.tight_layout()
plt.savefig('/home/leonidas/projects/itmo/optimal-control-theory/
   lab3/images/task3/states.png', dpi=200)

plt.figure(figsize=(6,4))
plt.plot(t, u)
plt.xlabel('t'); plt.ylabel('u'); plt.grid(True, alpha=0.3)
plt.title('Control u(t)')
plt.tight_layout()
plt.savefig('/home/leonidas/projects/itmo/optimal-control-theory/
   lab3/images/task3/u.png', dpi=200)

plt.figure(figsize=(6,4))
plt.plot(t, Jt)
plt.xlabel('t'); plt.ylabel('J(0,t)'); plt.grid(True, alpha=0.3)
plt.title('Accumulated cost J')
plt.tight_layout()
plt.savefig('/home/leonidas/projects/itmo/optimal-control-theory/
   lab3/images/task3/J.png', dpi=200)

print({'J_final': float(Jt[-1])})
```




(а) Состояния $x_1, x_2$




(б) Управление $u(t)$




(в) Накопленный критерий $J(0,t)$

Рисунок 1 — LQR, вариант 13: моделирование замкнутой системы

Для исходных параметров получено: $K \approx [1.1345\ 1.8229]$, собственные значения $A - bK$ равны $-1.17$ и $-9.03$ (устойчиво). Численно $J(\infty) \approx 2.419$.