

ETH Lectures 401-0663-00L Numerical Methods for Computer Science
401-2673-00L Numerical Methods for CSE

Homework Problems and Projects

Prof. R. Hiptmair, SAM, ETH Zurich

Autumn Term 2021, Version of October 2, 2022
(C) Seminar für Angewandte Mathematik, ETH Zürich

[Link](#) to the current version of this homework collection

0.1 General Information

0.1.1 Weekly Homework Assignments

All problems will be published in this single “.pdf” file. Every week, we publish a list of problem numbers on the lecture Moodle page. These are the current assignments for that week. Details are given on the lecture Moodle page.

0.1.2 Importance of Homework

Homework assignments are **not** mandatory. However, it is **very important** that you constantly exercise with the material you learn. “Solving” the homework assignments one week before the main exam by looking at the solutions will likely result in failure.

We provide hints and solutions from the beginning. It is your responsibility to look at the solutions only if you are stuck with a difficult problem and you tried to find a solution for a sufficient amount of time.

Most homework problems come with an estimate for the time required for its solution. These times are what we expect a bright and well-prepared student to require for that problem during an exam.

0.1.3 Corrections and Grading of Assignments

Homework assignments will not be graded.

However, you may submit your handwritten solutions by handing them to your tutor directly or uploading them via the Moodle upload interface, see the lecture Moodle page for instructions. The assistants will

Problem 6-8: Monomial representation of Chebychev polynomials

Chebychev polynomials are important both for theoretical analysis and algorithm design. For instance, as we have seen in [Lecture → Section 6.2.3.3] Chebyshev interpolants should internally be represented through their Chebychev expansion. In this problem we will study an algorithm for converting Chebychev expansions into monomial expansions.

This problem relies about elementary facts about Chebychev polynomials as introduced in [Lecture → Section 6.2.3.1].

On $[-1, 1]$ the n -th Chebychev polynomial T_n , $n \in \mathbb{N}_0$ is defined as [Lecture → Def. 6.2.3.3]

$$T_n(t) := \cos(n \arccos t) \quad -1 \leq t \leq 1. \quad (6.8.1)$$

That T_n is a polynomial of degree $\leq n$ is clear from the **3-term recursion**

$$T_{n+1}(t) = 2t T_n(t) - T_{n-1}(t) \quad , \quad T_0 \equiv 1, \quad T_1(t) = t, \quad n \in \mathbb{N}. \quad [\text{Lecture} \rightarrow \text{Eq. (6.2.3.5)}]$$

The set $\{T_0, \dots, T_n\}$ forms a basis of $\mathcal{P}_n$.

(6-8.a) 🕒 (8 min.) Below is an incomplete implementation of a C++ function

```
Eigen::VectorXd chebexpToMonom(const Eigen::VectorXd &a);
```

that expects the coefficients $a_j \in \mathbb{R}$, $j = 0, \dots, n$, of the Chebyshev expansion

$$p(t) = \sum_{j=0}^n a_j T_j(t), \quad t \in \mathbb{R},$$

of a polynomial $p \in \mathcal{P}_n$ to be passed in the vector $\mathbf{a}$ and returns the coefficients r_j , $j = 0, \dots, n$ with respect to the monomial basis,

$$p(t) = r_0 + r_1 t + r_2 t^2 + \dots + r_n t^n,$$

collected in a vector $[r_0, r_1, \dots, r_n]^\top$.

Fill in the missing pieces of the following code, which relies on [Lecture → Eq. (6.2.3.5)].

```
Eigen::VectorXd chebexpToMonom(const Eigen::VectorXd &a) {
    const int n = a.size() - 1; // degree of polynomial
    assert(n >= 0);
    Eigen::VectorXd r{ Eigen::VectorXd::Zero(n+1) };
    r[0] = A ;
    if (n > 0) {
        r[1] = B ;
        if (n > 1) {
            // two vectors temporarily storing the monomial coefficients
            // of Chebychev polynomials of two consecutive degrees
            std::array<Eigen::VectorXd, 2> c
            { Eigen::VectorXd::Zero(n+1), Eigen::VectorXd::Zero(n+1) };
            c[0][0] = 1.0; c[1][1] = 1.0;
            for (int j=2; j<=C; ++j) {
                c[j%2][0] *= -1.0;
```

```

    for (int k=D; k <= j; ++k)
        c[j%2][k] = E * c[F][G] - c[j%2][k];
    r += H * c[j%2];
}
}
}
return r;
}

```

- **A** $\triangleq$
- **B** $\triangleq$
- **C** $\triangleq$
- **D** $\triangleq$
- **E** $\triangleq$
- **F** $\triangleq$
- **G** $\triangleq$
- **H** $\triangleq$

SOLUTION of (6-8.a):

C++ code 6.8.2: Implementation of a function providing the monomial coefficients of a Chebychev expansion

```

1 Eigen::VectorXd chebexpToMonom(const Eigen::VectorXd &a) {
2     const int n = a.size() - 1; // degree of polynomial
3     assert(n >= 0);
4     // Vector for returning monomial coefficients
5     Eigen::VectorXd r{ Eigen::VectorXd::Zero(n+1) };
6     r[0] = a[0]; //  $T_0 \equiv 1$ 
7     if (n > 0) {
8         r[1] = a[1]; //  $T_1(t) = t$ 
9         if (n > 1) {
10             // two vectors temporarily storing the monomial coefficients
11             // of Chebychev polynomials of two consecutive degrees
12             std::array<Eigen::VectorXd, 2> c
13                 { Eigen::VectorXd::Zero(n+1), Eigen::VectorXd::Zero(n+1) };
14             c[0][0] = 1.0; c[1][1] = 1.0; // For  $T_0$  and  $T_1$ 
15             // Run through all Chebychev polynomials
16             for (int j=2; j<=n; ++j) {
17                 // Update monomial coefficients by 3-term recursion
18                 c[j%2][0] *= -1.0;
19                 for (int k=1; k <= j; ++k) {
20                     // Formula for 3-term recursion
21                     c[j%2][k] = 2*c[(j+1)%2][k-1] - c[j%2][k];
22                 }
23                 // Add monomial coefficients of contribution of  $T_j$ 
24                 r += a[j]*c[j%2];
25             }
26         }
27     }
28     return r;
29 }

```



End Problem 6-8 , 8 min.