Prof. R. Hiptmair
G. Alberti,
F. Leonardi

AS 2015

ETH Zürich
D-MATH

Numerical Methods for CSE

Problem Sheet 11

Problem 1 Efficient quadrature of singular integrands (core problem)

This problem deals with efficient numerical quadrature of non-smooth integrands with a special structure. Before you tackle this problem, read about regularization of integrands by transformation [1, Rem. 5.3.45].

Our task is to develop quadrature formulas for integrals of the form:

$$W(f) := \int_{-1}^{1} \sqrt{1-t^2}\, f(t)dt, \tag{34}$$

where f possesses an analytic extension to a complex neighbourhood of $[-1,1]$.

(1a) ⊡ The provided function

```
QuadRule gauleg(unsigned int n);
```

returns a structure `QuadRule` containing nodes (x_j) and weights (w_j) of a Gauss-Legendre quadrature (→ [1, Def. 5.3.28]) on $[-1,1]$ with n nodes. Have a look at the file `gauleg.hpp` and `gauleg.cpp`, and understand how the implementation works and how to use it.

HINT: Learn/remember how linking works in C++. To use the function `gauleg` (declared in `gauleg.hpp` and defined in `gauleg.cpp`) in a file `file.cpp`, first `include` the header file `gauleg.hpp` in the file `file.cpp`, and then compile and link the files `gauleg.cpp` and `file.cpp`. Using `gcc`:

```
g++ [compiler opts.] -c gauleg.cpp
g++ [compiler opts.] -c file.cpp
g++ [compiler opts.]  gauleg.o file.o -o exec_name
```

If you want to use CMake, have a look at the file CMakeLists.txt.

(1b) Study [1, § 5.3.37] in order to learn about the convergence of Gauss-Legendre quadrature.

(1c) Based on the function gauleg, implement a C++ function

```
template <class func>
double quadsingint(func&& f, unsigned int n);
```

that approximately evaluates (34) using $2n$ evaluations of f. An object of type func must provide an evaluation operator

```
double operator(double t) const;
```

For the quadrature error asymptotic exponential convergence to zero for $n \to \infty$ must be ensured by your function.

HINT: A C++ lambda function provides such operator.

HINT: You may use the classical binomial formula $\sqrt{1-t^2} = \sqrt{1-t}\sqrt{1+t}$.

HINT: You can use the template quadsingint_template.cpp.

(1d) Give formulas for the nodes c_j and weights $\tilde{w}_j$ of a $2n$-point quadrature rule on $[-1, 1]$, whose application to the integrand f will produce the same results as the function quadsingint that you implemented in (1c).

(1e) Tabulate the quadrature error:

$$|W(f) - \texttt{quadsingint(f,n)}|$$

for $f(t) := \frac{1}{2+\exp(3t)}$ and $n = 1, 2, ..., 25$. Estimate the $0 < q < 1$ in the decay law of exponential convergence, see [1, Def. 4.1.31].

Problem 2 Nested numerical quadrature

A laser beam has intensity

$$I(x, y) = \exp(-\alpha((x-p)^2 + (y-q)^2))$$

on the plane orthogonal to the direction of the beam.

(2a) Write down the radiant power absorbed by the triangle

$$\triangle := \{(x,y)^T \in \mathbb{R}^2 \mid x \geq 0, y \geq 0, x+y \leq 1\}$$

as a double integral.

HINT: The radiant power absorbed by a surface is the integral of the intensity over the surface.

(2b) Write a C++ function

```
template <class func>
double evalgaussquad(double a, double b, func&& f, const
    QuadRule & Q);
```

that evaluates an the N-point quadrature for an integrand passed in `f` in $[a, b]$. It should rely on the quadrature rule on the reference interval $[-1, 1]$ that supplied through an object of type `QuadRule`. (The vectors `weights` and `nodes` denote the weights and nodes of the reference quadrature rule respectively.)

HINT: Use the function `gauleg` declared in `gauleg.hpp` and defined in `gauleg.cpp` to compute nodes and weights in $[-1, 1]$. See Problem 1 for further explanations.

HINT: You can use the template `laserquad_template.cpp`.

(2c) Write a C++ function

```
template <class func>
double gaussquadtriangle(func&& f, int N)
```

for the computation of the integral

$$\int_{\triangle} f(x,y)dxdy, \tag{35}$$

using nested N-nodes, 1D Gauss quadratures (using the functions `evalgaussquad` of (2b) and `gauleg`).

HINT: Write (35) explicitly as a double integral. Take particular care to correctly find the intervals of integration.

HINT: Lambda functions of C++ are well suited for this kind of implementation.

(2d) ⚀ Apply the function `gaussquadtriangle` of (2c) to the subproblem (2a) using the parameter $\alpha = 1, p = 0, q = 0$. Compute the error w.r.t to the number of nodes N. What kind of convergence do you observe? Explain the result.

HINT: Use the "exact" value of the integral 0.366046550000405.

Problem 3 Weighted Gauss quadrature

The development of an alternative quadrature formula for (34) relies on the Chebyshev polynomials of the second kind U_n, defined as

$$U_n(t) = \frac{\sin((n+1)\arccos t)}{\sin(\arccos t)}, \qquad n \in \mathbb{N}.$$

Recall the role of the orthogonal Legendre polynomials in the derivation and definition of Gauss-Legendre quadrature rules (see [1, § 5.3.25]).

As regards the integral (34), this role is played by the U_n, which are orthogonal polynomials with respect to a weighted L^2 inner product, see [1, Eq. (4.2.20)], with weight given by $w(\tau) = \sqrt{1-\tau^2}$.

(3a) ⚁ Show that the U_n satisfy the 3-term recursion

$$U_{n+1}(t) = 2tU_n(t) - U_{n-1}(t), \qquad U_0(t) = 1, \qquad U_1(t) = 2t,$$

for every $n \geq 1$.

(3b) ⚀ Show that $U_n \in \mathcal{P}_n$ with leading coefficient 2^n.

(3c) ⚁ Show that for every $m, n \in \mathbb{N}_0$ we have

$$\int_{-1}^{1} \sqrt{1-t^2}\, U_m(t) U_n(t)\, dt = \frac{\pi}{2}\delta_{mn}.$$

(3d) ⚀ What are the zeros ξ_j^n ($j = 1, \ldots, n$) of U_n, $n \geq 1$? Give an explicit formula similar to the formula for the Chebyshev nodes in $[-1, 1]$.

(3e) ⚃ Show that the choice of weights

$$w_j = \frac{\pi}{n+1} \sin^2\left(\frac{j}{n+1}\pi\right), \qquad j = 1, \ldots, n,$$

ensures that the quadrature formula

$$Q_n^U(f) = \sum_{j=1}^{n} w_j f(\xi_j^n) \tag{36}$$

provides the exact value of (34) for $f \in \mathcal{P}_{n-1}$ (assuming exact arithmetic).

HINT: Use all the previous subproblems.

(3f) ⚁ Show that the quadrature formula (36) gives the exact value of (34) even for every $f \in \mathcal{P}_{2n-1}$.

HINT: See [1, Thm. 5.3.21].

(3g) ⚂ Show that the quadrature error

$$|Q_n^U(f) - W(f)|$$

decays to 0 exponentially as $n \to \infty$ for every $f \in C^\infty([-1,1])$ that admits an analytic extension to an open subset of the complex plane.

HINT: See [1, § 5.3.37].

(3h) ⚁ Write a C++ function

```
template<typename Function>
double quadU(const Function &f, unsigned int n)
```

that gives $Q_n^U(f)$ as output, where `f` is an object with an evaluation operator, like a lambda function, representing f, e.g.

```
auto f = [] (double & t) {return 1/(2 + exp(3*t));};
```

(3i) ⚁ Test your implementation with the function $f(t) = 1/(2+e^{3t})$ and $n = 1,\dots,25$. Tabulate the quadrature error $E_n(f) = |W(f) - Q_n^U(f)|$ using the "exact" value $W(f) = 0.483296828976607$. Estimate the parameter $0 \le q < 1$ in the asymptotic decay law $E_n(f) \approx C q^n$ characterizing (sharp) exponential convergence, see [1, Def. 4.1.31].

Problem 4 Generalize “Hermite-type” quadrature formula

(4a) ⚃ Determine $A, B, C, x_1 \in \mathbb{R}$ such that the quadrature formula:

$$\int_0^1 f(x)dx \approx Af(0) + Bf'(0) + Cf(x_1) \tag{37}$$

is exact for polynomials of highest possible degree.

(4b) ⚃ Compute an approximation of $z(2)$, where the function z is defined as the solution of the initial value problem

$$z'(t) = \frac{t}{1+t^2} \quad , \quad z(1) = 1 \; . \tag{38}$$

Issue date: 26.11.2015

Hand-in: 03.12.2015 (in the boxes in front of HG G 53/54).

Version compiled on: November 26, 2015 (v. 1.0).

References

[1] R. Hiptmair. *Lecture slides for course "Numerical Methods for CSE"*. http://www.sam.math.ethz.ch/~hiptmair/tmp/NumCSE/NumCSE15.pdf. 2015.