Numerical Algorithms

Winter 2023/24
Prof. Dr. Carsten Burstedde
Hannes Brandt

Exercise Sheet 7.

Submission: -

Exercise 1. (Programming Task II)

On a 2D triangle mesh described by a `.node` and an `.ele` output file of the `Triangle` library we define a piecewise linear nodal basis $\{\phi_i\}_{i=1,\dots,\texttt{\#nodes}}$, where ϕ_i is 1 on node i, 0 on all other nodes and affine linear on every triangle of the triangulation.

We want to lay the necessary foundations to implement a solver of the Laplace equation in the third and final programming exercise. To achieve this, we need to be able to compute the matrix-vector product for the mass matrix M and the stiffness matrix A of the basis $\{\phi_i\}_{i=1,\dots,\texttt{\#nodes}}$. For our basis of hat functions, both M and S will be sparse and we want to avoid to compute and store them explicitly.

Instead, we can compute the matrix-vector products in an element-wise approach. We loop over all triangles of the mesh and add their contribution to the matrix-vector product onto the result vector one after another. Every triangle T_e only affects the three nodal basis functions of its corners. All other basis functions are all-zero on T_e. The triangles impact can be described by 3×3 local matrices M^e and A^e.

To compute M^e and A^e we consider the reference triangle T_0 spanned by the corners $(0,0), (1,0)$ and $(0,1)$. For each triangle T_e of the mesh we can find an affine linear mapping

$$F_e\colon \mathbb{R}^2 \to \mathbb{R}^2; x \mapsto Bx + x_0 \qquad \text{with } B \in \mathbb{R}^{2\times 2}, x_0 \in \mathbb{R}^2$$

with $T_e = F_e(T_0)$. Based on this linear mapping we can compute M^e using the reference local matrix M^0

$$M^e_{ij} = |DF_e| \int_{T_0} \phi_i\phi_j = |DF_e|\, M^0_{ij}. \tag{1}$$

With a similar approach we obtain

$$A^e_{ij} = |DF_e| \int_{T_0} \nabla\phi_i\,(DF_e)^{-1}\,(DF_e)^{-T}\,\nabla\phi_j \tag{2}$$

for the stiffness matrix of the Laplace equation. So, it suffices to do most of the computation only once for the reference element T_0.

For verification of the mass-matrix-vector product, we define $e := (1,\dots,1) \in \mathbb{R}^{\#\text{nodes}}$. In our basis $\{\phi_i\}_{i=1,\dots,\texttt{\#nodes}}$ the vector e corresponds to the function that is constant 1 on the whole mesh. Therefore, the product $e^T M e$ should be equal to the area of the domain covered by the mesh. For verification of the stiffness-matrix-vector product, we can compute Ae, which should be 0, since the function is constant on the whole mesh.

a) Verify the equations (1) and (2) using the transformation theorem.

b) Construct an affine linear mapping F_e that maps the reference element T_0 onto an arbitrary triangle T_e with corners $a, b, c \in \mathbb{R}^2$.

c) Based on the element-iterator from the first programming exercise and the mapping from b), implement the mass-matrix-vector product and the stiffness-matrix-vector product. Use the equations (1) and (2) to compute M^e and A^e. The matrix M^0 and the gradients $\nabla\phi_i$ may be hardcoded and should be set/computed only once.

d) Verify the correctness of your mass-matrix-vector product. Based on your results from the first programming exercise, load the `box.4.node` and `box.4.ele` example files available on the Florida State University site, which was linked on the lectures website. Here, you may ignore the third coordinate of the nodes and consider the mesh as a simple 2D mesh. Compute $e^T Me$ and compare it to the mesh's expected area of 8.

   As an additional verification, you may also load the `greenland.node` and `greenland.ele` files from same source, which contain a triangulation of Greenland in a scale of approximately 1 : 4.5. Compare your results with the actual area of Greenland. You should have an error margin of about 10%. Again, you may ignore the third coordinate of the nodes in your calculations.

e) Ensure basic correctness of your stiffness matrix-vector product by checking Ae.

(0 Points)