`reparam_cm` - Short Documentation

B. Stinner
Mathematics Institute, Zeeman Building
University of Warwick
UNITED KINGDOM

Abstract

Introduction

Given a two-dimensional mesh Γ_h with N vertices `x_dh->vec[`i`]`, $i = 0, \ldots, N-1$, the goal is to rearrange the grid points, typically in order to obtain a nicer mesh. The idea is to compute a conformal map ϕ_h from Γ_h to the sphere S^2 with the method described in [1] and to re-project a nice mesh with vertices `y_dh->vec[`i`]`, $i = 0, \ldots, N-1$, given on S^2 to Γ_h.
Based on ALBERTA-routines, the function `reparam_cm (DOF_REAL_D_VEC *y_dh, DOF_REAL_D_VEC *x_dh, struct reparam_cm_data *crd)` performs the following steps:

1. compute a conformal map $\phi_h : \Gamma_h \to S^2$, i.e., a minimiser of the Dirichlet energy
$$D(\phi_h) := \int_{\Gamma_h} \frac{1}{2} |\nabla_{\Gamma_h} \phi_h|^2,$$
2. project the vertices of a given nice mesh on S^2 to the induced mesh $\phi_h(\Gamma_h)$ on S^2,
3. create a new mesh of Γ_h by applying ϕ_h^{-1} to the projected vertices and compute the values of other fields, stored in `crd->drvl, crd->drdvl`, on Γ_h in the new vertices.

Restrictions:

- two-dimensional meshes of sphere topology in three-dimensional space,
- linear finite element functions,
- UZAWA solver contains some bug; use GMRES solver,
- the mesh on S^2 induced by `y_dh` must be oriented such that the normal n on a triangle with vertices p_0, p_1, p_2 (ordered in this way), computed via $n = (p_2 - p_1) \times (p_0 - p_2)$, points away from the origin, i.e., $n \cdot p_i > 0$ for all i.

Some notation

The vertices of Γ_h also are denoted by $\boldsymbol{x}_i :=$ `x_dh->vec[i]`, $i = 0, \ldots, N-1$. Linear finite elements are considered on Γ_h, i.e., the spaces

$$V_h := \big\{ v \in C^0(\Gamma_h, \mathbb{R}) \,\big|\, v|_T \in \mathcal{P}^1(T, \mathbb{R}) \text{ for all triangles } T \in \Gamma_h \big\}, \quad \boldsymbol{V}_h := V_h^3.$$

The standard basis function of V_h are denoted by $\{b_i\}_{i=0}^{N-1}$, a basis of $\boldsymbol{V}_h$ then consists of the functions $\{\boldsymbol{e}_k b_i\}_{i,k=0}^{N,2}$ where the $\boldsymbol{e}_k$ are the standard basis of $\mathbb{R}^3$. For a function $\phi_h \in \boldsymbol{V}_h$ we introduce the notation

$$\phi_{ik} := \phi_h(\boldsymbol{x}_i) \cdot \boldsymbol{e}_k, \quad \underline{\phi}_k := (\phi_{ik})_{i=0}^{N-1}, \quad \underline{\phi} := (\phi_{ik})_{i,k=0}^{N-1,2}, \quad \phi_i := (\phi_{ik})_{k=0}^{2}.$$

By the vertices y_dh->vec[i], $i = 0, \ldots, N-1$, and the topology given by Γ_h a mesh is induced on S_2. We write S_h^2 for this discretised sphere. A parameterisation $u_h : S_h^2 \to \Gamma_h$ is defined by mapping y_dh->vec[i] to x_dh->vec[i], $i = 0, \ldots, N-1$.

Computation of the conformal map

Essentially explained in [1]. The goal is to minimise $D(\phi_h)$ subject to constraints as follows. First, we want that

$$\phi_i \in S^2, \quad i = 0, \ldots, N-1.$$

Further constraints occur since harmonic maps are unique only up to the conformal group of S^2. Let

$$M_l(\phi_h) := \boldsymbol{e}_l \cdot \int_{S_h^2} (\phi_h \circ u_h)(\boldsymbol{y}) d\mathcal{H}^2(\boldsymbol{y}), \quad l = 0, 1, 2,$$

$$M_l(\phi_h) := \int_{S_h^2} (\phi_h \circ u_h)(\boldsymbol{y}) \cdot \boldsymbol{Z}_l \boldsymbol{y} d\mathcal{H}^2(\boldsymbol{y}), \quad l = 3, 4, 5$$

where the matrices

$$\boldsymbol{Z}_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \boldsymbol{Z}_4 = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, \quad \boldsymbol{Z}_5 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$$

are a basis of the skew symmetric real 3×3 matrices. Let us introduce the vectors

$$\underline{m^l} := \{M_l(\boldsymbol{e}_k b_i)\}_{i,k=0}^{N-1,2}, \quad l = 0, \ldots, 5.$$

The additional constraints then read

$$0 = M_l(\phi_h) = \sum_{i,k=0}^{N-1,2} m_{ik}^l \phi_{ik} = \underline{m^l} \cdot \underline{\phi}, \quad l = 0, \ldots, 5.$$

Remark: Those linear constraints depend on the manifold to which ϕ maps. E.g., there are less constraints for a torus.

The constraints are taken into account with Lagrange multipliers $\lambda = (\lambda_0, \ldots, \lambda_{N-1}) \in \mathbb{R}^N$ and $\rho = (\rho_0, \ldots, \rho_5) \in \mathbb{R}^6$. We define

$$F(\underline{\phi}, \lambda, \rho) := \frac{1}{2} \underline{\phi} \cdot \boldsymbol{A} \underline{\phi} + \frac{1}{2} \sum_{i=0}^{N-1} \lambda_i \big(|\phi_i|^2 - 1\big) + \sum_{l=0}^{5} \rho_l \underline{m^l} \cdot \underline{\phi}.$$

Now, the target is to solve

$$F'(\underline{\phi}, \lambda, \rho) = 0$$

with a Newton method. The first and second derivative of F are stated below. To compute the new search direction

$$d^m := (F''(\underline{\phi}^m, \lambda^m, \rho^m))^{-1} F'(\underline{\phi}^m, \lambda^m, \rho^m)$$

we have to solve a linear system with a matrix of saddle point structure. GMRES is a possible method. Data for this solver must be provided in the structure crd->spp_gmres_data. Data for the Newton method as tolerance, maximal iteration number etc. in the structure crd->ns_data.

Here are the formulas for the derivatives of F:

$$
\begin{aligned}
F_{,\underline{\phi}}(\underline{\phi},\lambda,\rho) &= \boldsymbol{A}\underline{\phi} + \big(\lambda_i\phi_{i,k}\big)_{i,k=0}^{N-1,2} + \sum_{l=0}^{5}\rho_l\cdot\underline{m^l},\\
F_{,\lambda}(\underline{\phi},\lambda,\rho) &= \frac{1}{2}\sum_{i=0}^{N-1}\big(|\phi_i|^2-1\big),\\
F_{,\rho}(\underline{\phi},\lambda,\rho) &= \big(\underline{m^l}\cdot\underline{\phi}\big)_{l=0}^{5},\\
F_{,\underline{\phi}\underline{\phi}}(\underline{\phi},\lambda,\rho) &= \boldsymbol{A} + \begin{pmatrix}\mathrm{diag}(\lambda) & & \\ & \mathrm{diag}(\lambda) & \\ & & \mathrm{diag}(\lambda)\end{pmatrix},\\
F_{,\underline{\phi}\lambda}(\underline{\phi},\lambda,\rho) &= \begin{pmatrix}\mathrm{diag}(\underline{\phi}_0)\\ \mathrm{diag}(\underline{\phi}_1)\\ \mathrm{diag}(\underline{\phi}_2)\end{pmatrix} = F_{,\lambda\underline{\phi}}(\underline{\phi},\lambda,\rho)^T,\\
F_{,\underline{\phi}\rho}(\underline{\phi},\lambda,\rho) &= \big(\underline{m^0},\dots,\underline{m^5}\big) = F_{,\rho\underline{\phi}}(\underline{\phi},\lambda,\rho)^T,\\
F''(\underline{\phi},\lambda,\rho) &= \begin{pmatrix}
A+\mathrm{diag}(\lambda) & 0 & 0 & \mathrm{diag}(\underline{\phi}_0) & | & & | \\
0 & A+\mathrm{diag}(\lambda) & 0 & \mathrm{diag}(\underline{\phi}_1) & \underline{m^0} & \cdots & \underline{m^5} \\
0 & 0 & A+\mathrm{diag}(\lambda) & \mathrm{diag}(\underline{\phi}_2) & | & & | \\
\mathrm{diag}(\underline{\phi}_0) & \mathrm{diag}(\underline{\phi}_1) & \mathrm{diag}(\underline{\phi}_2) & 0 & \cdots & \cdots & 0 \\
— & (\underline{m^0})^T & — & \vdots & \ddots & & \vdots \\
 & \vdots & & \vdots & & \ddots & \vdots \\
— & (\underline{m^5})^T & — & 0 & \cdots & \cdots & 0
\end{pmatrix}
\end{aligned}
$$

Sometimes, blocks of dimension $N\times N$ have been used, and the dimension index arranges the ordering of the blocks. The following abbreviations were used:

$$
\begin{aligned}
\boldsymbol{A} &= \big(\delta_{kl}A_{ij}\big)_{i,k,j,l=0}^{N-1,2,N-1,2}, \quad A_{ij} = \int_{\Gamma_h}\nabla_{\Gamma_h}b_i\cdot\nabla_{\Gamma_h}b_j,\\
\mathrm{diag}(\lambda) &= \big(\delta_{ij}\lambda_i\big)_{i,j=0}^{N-1,N-1},\\
\mathrm{diag}(\underline{\phi}_k) &= \big(\delta_{ij}\phi_{ik}\big)_{i,j=0}^{N-1,N-1}.
\end{aligned}
$$

Vertex projection

The new mesh on Γ_h is defined by re-projecting the vertices `y_dh->vec[i]` with ϕ_h^{-1} from S_h^2 to Γ_h. For this purpose, the vertices must be projected to the mesh $\phi_h(\Gamma_h)$ induced by ϕ_h on S^2:

1. **Projections to close triangles:** First, run over all elements $T\in\phi_h(\Gamma_h)$. For a triangle T denote the vertices with ϕ_h`(x_dh->vec[`i_l`])`, $l=0,1,2$. Now, check whether the orthogonal projections pr_{i_l} of the `y_dh->vec[`i_l`]` to the plane in which T lies are indeed on T. If this is the case for, let's say, pr_{i_l} then
   - compute the barycentric coordinates and store them in `barcor[`k`]->vec[`i_l`]`, $k=0,1,2$,
   - store the corresponding global indices i_k in `gdof[`k`]->vec[`i_l`]`, $k=0,1,2$,
   - set `nbc->vec[`i_k`]` $=3$, i.e., the number of barycentric coordinates.
2. **Projections to other triangles:** Second, run again over all elements $T\in\phi_h(\Gamma_h)$. For each triangle T now run over all (not yet projected) vertices `y_dh->vec[i]` and try to project them to T as described in the former point. Since we consider meshes on the sphere there may be triangles onto which an orthogonal projection is possible. To select the right one the scalar product of the vertex with the *external* unit normal of T with respect to the sphere is checked to be positive. *Here, we need the constraint on the orientation of the mesh induced by* `y_dh` *on* S^2.

3. **Projection to edges:** There might still be unprojected vertices lying over edges or other vertices of $\phi_h(\Gamma_h)$. A third run over all not yet projected vertices `y_dh->vec[`i`]` is performed. For each such vertex (w.l.o.g. with index i) a loop over the triangles $T \in \phi_h(\Gamma_h)$ is performed. The distance of `y_dh->vec[`i`]` to each edge of T is computed. This way, the closest edge is found. Finally,
   - compute the barycentric coordinates with respect to the closest edge and store them in `barcor[`k`]->vec[`i`]`, $k = 0, 1$,
   - store the global indices i_0, i_1 corresponding to the vertices defining this edge in `gdof[`k`]->vec[`i`]`, $k = 0, 1$,
   - set `nbc->vec[`i_k`]` $= 2$ (number of barycentric coordinates).

The field `nbc`, initialised with zero, is also used as a flag field to indicate whether a vertex has already been projected or not.

Creation of the new mesh

The fields `nbc, barcor, gdof` computed during the projection are used to obtain a new mesh on Γ_h as follows. Observe that the projection pr_i of `y_dh->vec[`i`]`, $i \in \{0, \dots, N-1\}$, to $\phi_h(\Gamma_h)$ is

$$pr_i = \sum_{k=0}^{\texttt{nbc->vec[}i\texttt{]}} \texttt{barcor[}k\texttt{]->vec[}i\texttt{]}\ \phi_h\big(\texttt{x\_dh->vec[gdof[}k\texttt{]->vec[}i\texttt{]]}\big).$$

Applying ϕ_h^{-1} yields *because of the linearity of ϕ_h on each triangle*

$$\texttt{xnew\_dh->vec[}i\texttt{]} := \phi_h^{-1}\big(pr_i\big) = \sum_{k=0}^{\texttt{nbc->vec[}i\texttt{]}} \texttt{barcor[}k\texttt{]->vec[}i\texttt{]}\ \texttt{x\_dh->vec[gdof[}k\texttt{]->vec[}i\texttt{]]}.$$

Other fields belonging to V_h or $\boldsymbol{V}_h$ can be adapted similar to `x_dh`. Exemplary, for `s_h` $\in V_h$:

$$\texttt{snew\_h->vec[}i\texttt{]} := \sum_{k=0}^{\texttt{nbc->vec[}i\texttt{]}} \texttt{barcor[}k\texttt{]->vec[}i\texttt{]}\ \texttt{sold\_dh->vec[gdof[}k\texttt{]->vec[}i\texttt{]]}.$$

References

[1] G. Dziuk, U. Clarenz, *Numerical methods for conformally parametrized surfaces*, CPDw04 - Interphase 2003: Numerical Methods for Free Boundary Problems, workshop at the Isaac Newton Institute (2003),
`http://www.newton.cam.ac.uk/webseminars/pg+ws/2003/cpd/cpdw04/0415/dziuk/`